
NOVIEMBRE 1987
Revista para usuarios de
Drean
COMMODORE
AÑO 2 Nº 23 A 5 REP. ARGENTINA
PROGRAMACION
COMO HACER
UNA BATALLA NAVAL
PASO A PASO
DEFENDER OF THE CROWN
TRUCOS Y RUTINAS
UTILES
CONCURSO
MENSUAL
ULTIMO MES
NUEVO SORTEO
RANKING DE SOFTWARE

SUMARIO

NOTAS TECNICAS

PROGRAMAS

REVISION DE SOFT

NOTICIAS

SECCIONES FIJAS



Desarrollamos la forma en que podemos hacer que la máquina "piense". El juego que tratamos esta vez es la batalla naval.



Mostramos las ventajas del C como lenguaje estructurado y cuál es la mejor forma de acercarse al mismo.



Para aquellos que quieren que los juegos de video sean algo más que una matanza de extraterrestres o una carrera de obstáculos, "Defender of the Crown" llega para lanzar un desafío: luchar por la corona y unificar el reino bajo el mando de un sajón.

Drean Commodore AÑO 2 Nº 23 1987

Director General
Ernesto del Castillo

Director Editorial
Cristián Pusso

Director Periodístico
Fernando Flores

Secretario de Redacción
Ariel Testori

Arte y Diagramación
Fernando Amengual
Tamara Migelson

Fotografía
Víctor Grubsey
Image Bank
Eduardo Comesaña

Departamento de Avisos
Oscar Devoto
Nelso Capello

Revista para usuarios de Drean Commodore es una publicación mensual editada por editorial **PROEDI** S.A., Paraná 720, 5º Piso(1017) Buenos Aires. Tel.: 46-2886 y 49-7130. Reg. Nac. de la Prop. Intelectual E.T., M. Registrada. Precio de este ejemplar, ₳ 5 Los ejemplares atrasados se venderán al precio del último número en circulación. Queda hecho el depósito que indica la Ley 11.723 de Propiedad Intelectual ISSNO326-8233

Impresión: Calcotam. **Fotocromo tapa:** Columbia.

Distribuido en Capital: Martino, Juan de Garay 358, P.B. Capital. **Distribuidor interior:** DGP, Hipólito Yrigoyen 1450, Capital Federal. T.E.: 38-9266/9800.

NOTICIAS DREAN COMMODORE

CONTABILIDAD Y BANCOS

S.A.M. (Sistemas Administrativos Modernos) lanzó una serie de programas que distribuye en forma exclusiva **Poke Computación** . Los sistemas son los siguientes: de Contabilidad General (con 350 cuentas y 1200 asientos), de Sueldos y Jornales (con 80 legajos y 100 conceptos), y de Bancos (un sistema totalmente interactivo que permite a la mediana empresa obtener información confiable y ordenada que posibilitará un mayor control).

DATA BECKER CAMBIA SU IMAGEN



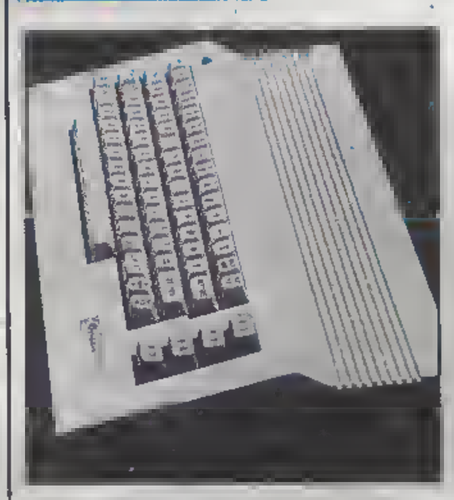
Sus libros siempre se ubicaron entre los más importantes, mejor escritos y los que abarcan más temas sobre computadoras existentes en el país. Pero eran también los más caros. Hoy, gracias a que **Data Becker** los comenzó a editar en la Argentina, están alcanzando un precio más accesible. El servicio local ya cuenta con 25 títulos sobre Commodore 64 y 12 sobre Commodore 128. Otra buena noticia referida a esta editorial es que también se ocupará de títulos técnicos de otras editoriales mundiales de gran prestigio de los cuales han adquirido los derechos para imprimir en nuestro país. El plan contempla la incorporación de 5 novedades mensuales.

PARAISO DE LOS JUEGOS



Los commodorianos podrán estar al tanto de lo último que sale, ponerse en contacto con quienes comparten las mismas actividades, tener acceso a programas especiales, y desarrollar cualquier programa de entretenimiento. Esa es la propuesta de **Microgame**. Que el usuario realmente entre en el "paraíso de los juegos".

SOFTWARE COMERCIAL



La empresa **Logitach Sistemas** lanzó una línea de software comercial para las computadoras Drean Commodore 64 y 128. Los programas son los siguientes: Mailing de clientes, Stock y proveedores, Compras, Ventas, Ventas y proveedores, Compras y ventas, Sueldos y jornales, Bancos, IVA compras y ventas y Lista de precios.

DATASOUND



Cuando un programa no "carga", puede deberse a varias causas. Por ejemplo, que la señal grabada en el casete sea demasiado débil en amplitud o que esté distorsionada en su forma, debido a la utilización de cintas de baja calidad, o bien al empleo de algún método casero de grabación, o a una combinación de ambas cosas.

Pero la principal causa de que muchos casetes no puedan cargarse es la distinta alineación del cabezal (¡el bendito cabezal!) del datasete que grabó originalmente el programa con la del que lo reproduce. Estas diferencias en las posiciones de los cabezales hace que la señal reproducida no sea copia fiel de la original y, por lo tanto, la computadora no la reconozca.

El DATASOUND, desarrollado por **JDC**, es una interfase que, intercalada entre la máquina y el datasete, nos permite escuchar en la T.V. la señal que ingresa a la computadora proveniente del casete. Esto nos permite contar con un excelente patrón de calibración.

El modo de uso es muy simple. Debemos colocar el casete que no carga normalmente y ajustar el cabezal de nuestro datasete hasta que la señal escuchada sea nítida y aguda. En ese punto, la alineación obtenida

será idéntica a la que se empleó para grabar el casete originalmente, y el programa cargará sin dificultad.
Viene con un programa, en casete o en disco, y una grabación en cinta que contiene una señal de referencia para la alineación del cabezal del datasete.
El DATASOUND es compatible con todas las marcas de datasete y además puede quedar conectado en forma permanente a la máquina, ya que no interfiere con la carga o grabación normal de los programas. (Fabrica y distribuye JDC computación.)

LANZAMIENTO MUNDIAL DE LA COMMODORE A2000

Este equipo viene a cubrir muchos de los defectos de la A1000 manteniendo sus ventajas, lo que la pone en un lugar muy importante para tener en cuenta a la hora de analizar la compra de un equipo de uso profesional.
Las primeras observaciones que podemos hacer surgen de la visualización externa del sistema. La A2000 viene en una "caja" del mismo ancho pero algunos centímetros más larga y alta que la A1000. El port del mouse fue trasladado del lado derecho de la consola a la parte trasera de la máquina.
En el frente del panel encontramos un drive para discos 3 1/2 y uno de 5 1/4 o, en su defecto, de hard disk. En cuanto a las especificaciones del hardware podemos notar similitudes con respecto a la A1000. El microprocesador Motorola 68000, que corre a 7,14 MHz; los mismos chips de video, sonido y DMA.
Pero luego la 2000 tiene otras características: un reloj/calendario, capacidad de memoria de 1 megabyte y 256 K de ROM. Contiene la última versión del sistema de rutinas de bajo nivel, y elimina de esta manera el "kickstart" o disco cargador de estas rutinas.
Los ports de serie y paralelo aceptan cables estándar. El teclado posee 94 teclas. Algunas de las nuevas son para "help", "enter" y las operaciones matemáticas en el teclado numérico. Las teclas de cursor tienen una disposición de T invertida. En el



modelo anterior la orientación era en forma de una cruz.

SLOTS

La A2000 está equipada con siete slots internos para los periféricos, en lugar de la "caja cerrada" que es la A1000, que tiene un solo conector externo.
El sistema del bus de la Amiga tiene 5 de esos slots y 4 están en el bus secundario, que es compatible con IBM XT (Commodore agregó además la posibilidad de una potencial compatibilidad con las IBM AT en el futuro).
A esta altura hay que hacer una aclaración importante, porque hablamos de 7 slots, que estaban distribuidos 5 en un bus y 4 en el otro (5+4=7?). No, no nos volvimos locos. La explicación está dada en que 4 slots son usados como puentes de comunicaciones Amiga/PC. La tarjeta A2088, por ejemplo, está en uno de esos puentes. Si se quiere emular a los PC, hay que ponerla de manera que nos queden 3 de los 4 bus de PC libres, o se la puede colocar para que permita usar 4 de los cinco de la Amiga.

BUSES

El "tráfico" de datos entre los slots y la CPU es manejado por tres buses diferentes. El bus de la CPU consiste en 86 líneas, el de PC en 62 y el Amiga en 100 líneas que llevan información a la CPU.

POSIBILIDADES

Aquellos que quieran trabajar con hard disk y una capacidad de memoria de 8 megabytes en una Amiga 2000 lo pueden hacer por la facilidad de expansión que tiene. Recordemos que, si bien a la 1000 era posible expandirla, esa tarea fue encarada por pequeñas compañías que trabajan con pequeños volúmenes y altos precios.
Commodore ha lanzado ahora, junto a su nuevo "bebé", una serie de periféricos de apoyo al mismo.
Los que se sintieron atraídos por las capacidades gráficas u otras características de la Amiga, pero sentían dudas en cuanto a la memoria disponible, no tienen más motivos de preocupación.
Lo mismo para aquellos que quieren un equipo compatible IBM PC, ahora disponen de esa capacidad en la Amiga 2000. El precio del equipo, U$S 1500 en Estados Unidos, tentó inclusive a muchos usuarios hogareños, que dieron el salto de las "home" a las PC.

JUEGOS DE ESTRATEGIA

Desarrollamos la forma de hacer que la máquina "piense". El juego que tratamos esta vez es la batalla naval.



Todos alguna vez, en un día de lluvia, en la clase de literatura de tercer año o en algún otro sitio donde nos hayamos aburrido, jugamos a la batalla naval. Los elemenos del juego, papel y lápiz, son fáciles de conseguir (siempre hay alguien que nos presta hojas cuadriculadas). Incluso los fabricantes locales de juegos han hecho versiones "de lujo" para los más fanáticos. Mi tía, una de esas tías que no faltan en ninguna familia, se lamenta de que los chicos hayan perdido esa clase de juegos con la llegada de la computadora. No pretendemos hacer aquí un estudio sociológico para refutar a mi tía. Mostraremos que la computadora puede no sólo suplir al papel y al lápiz sino que también nos puede hacer entender las estrategias para jugar mejor.

¿Cómo encarar un programa que juegue a la batalla naval? De la misma manera que todo programa. La primera herramienta que tiene el programador es su propio lenguaje. Con él debemos describir los pasos a seguir como si le estuviesemos enseñando a jugar a un amigo y luego tenemos que releer lo escrito analizándolo. Nos ponemos en el lugar de la máquina; lo ignoramos todo y esas instrucciones nos deben enseñar a jugar. De ahí seguramente saldrán acotaciones y nuevas instrucciones. Para esta etapa también se puede usar un grabador, de acuerdo con qué lenguaje maneje más comodamente el programador, sea el oral o el escrito.

En la batalla naval, por ejemplo, podemos hacer una primera explicación de esta manera: situar barcos propios - situar barcos enemigos - elegir jugada - respuesta del jugador - jugada del jugador - respuesta - repetir proceso desde elegir jugada.

De esta manera tenemos definidos algunos bloques y, a grandes rasgos, el proceso a seguir. En una segunda lectura nos damos cuenta que faltan cubrir ciertos detalles como hacer un control de victoria (para finalizar) luego de cada respuesta o determinar quién es el que comienza a jugar. Quedaría luego un proceso que, dividido en bloques, podemos listar (ver figura 1).

Una vez deteminado esto (que a manera general podemos extender a todos los juegos: Etapa inicial - Juega máquina - Juega hombre - Repite - Resultados) comenzamos a analizar rutina por rutina.

SE DISTRIBUYE LA FLOTA

Situar barcos propios: (subrutina 1000): Vamos a colocar un barco de 5 cuadrados (portaaviones), dos de 4 (cruceros), dos de 3 (acorazados) y cuatro de 2 (submarinos).

Para ello vamos a tomar primero los barcos más grandes y elegimos una casilla al azar, que tomamos como cabecera, para recién luego determinar la dirección hacia donde se extenderá el barco. Si hay más de una, se elige al azar. Una vez colocado el primer barco, para colocar los siguientes hay que tomar en cuenta la condición de que no se superpongan con los ya colocados.

Situar barcos enemigos (subrutina 2000): Hay que habilitar una matriz en donde el jugador pueda en-

Figura 1

```
10 gosub 1000: rem situa barcos propios
20 gosub 2000: rem situa barcos enemigos
30 gosub 3000: rem determina la mano
40 gosub 4000: rem elegir jugada
50 gosub 5000: rem respuesta del juador
60 gosub 6000: rem control de victoria de lA maquina
70 gosub 7000: rem jugador entra jugada
80 gosub 8000: rem respuesta de la maquina
90 gosub 9000: rem control de victoria del jugador
100 goto 40
```

trar a su antojo y colocar sus barcos. Luego hay que verificar que no haya superposiciones y que estén todos los barcos.

Determinar mano (subrutina 3000): Se elige si se quiere ser mano o no (por ejemplo M=0 o M=1). Luego se complementa con un cambiador de mano en el proceso central (M=1-M) y un distribuidor de mano para que juegue quien corresponda. Lo que hace el cambiador de mano es alternar los valores de 0 y 1 para que se juegue por turno.

COMIENZA LA BATALLA

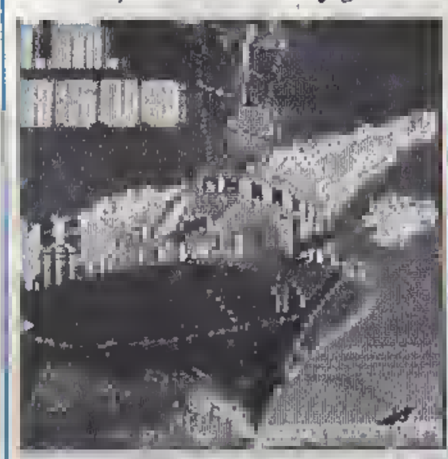
Elegir jugada (subrutina 4000): Esta es la parte donde hay que planear la estrategia. Acá inciden dos factores: uno, la suerte, que no es "programable". El otro surge del hecho de ver que gana aquel que cubre el tablero primero. Ahora bien, si se juega por turno, sería de esperar que ambos tableros se cubran por igual. Esto no es así, ya que por cada disparo acertado y barco hundido hay casillas que se cubren automáticamente por agua (dos barcos no pueden estar en casillas vecinas).

De esta manera los disparos acertados en el centro del tablero nos cubren cuatro casillas con agua, mientras que los de los bordes, solo dos, y los vértices, uno (ver figura). Cuantas más casillas podamos cubrir con agua, automáticamente más chances tendremos de ganar.

Podemos dividir la estrategia en dos partes, depende de si hay algún barco tocado o no. En el primer caso trataremos de hundir al barco dañado, mientras que en el segundo desarrollaremos la estrategia que hemos sugerido.

La casilla elegida se busca siguiendo las características mencionadas, y entre las diferentes opciones se selecciona una al azar. Es importante tener en cuenta esto para evitar juegos repetidos y para que el contrincante no sepa dónde ubicar sus barcos para tener mayores chances.

Respuesta del jugador (subrutina 5000): Determina si la jugada de



la máquina tocó algún buque, lo hundió, o dió en el agua. En el caso que haya dado en el blanco cubre con agua las casillas adyacentes en las que no puede haber barcos.

Control de victoria de la máquina (subrutina 6000): Se determina si ya están hundidos todos los barcos enemigos.

Jugada del hombre (subrutina 7000): Se pide la jugada y se verifica si es posible hacerla.

Respuesta de la máquina (subrutina 8000): Similar a la subrutina 5000. Incluso programando con cuidado es factible utilizar una sola rutina para ambos casos.

Control de victoria del hombre (subrutina 9000): Esta rutina controla si ya están hundidos todos los barcos de la máquina. Es similar a la subrutina 6000 y, al igual que a la anterior, podemos unificarla.

Ya están delineados los bloques y la estructura de programación que vamos a seguir. El próximo paso es ver los elementos que utilizaremos. Necesitamos dos matrices de 10 x 10 por jugador; una para la situación de los barcos propios y otra para almacenar los tiros que se van ejecutando y su resultado. Utilizaremos también vectores para barcos tocados y hundidos y para almacenar posibles respuestas.

Las variables dependerán de los gustos de cada uno, pero recomendamos hacer un esquema de cuales variables vamos a utilizar en todo el programa y cuáles en alguna rutina en particular. Así evitaremos la repetición de las mismas y los inconvenientes que ello implica.

Creemos que ahora están dadas las bases y herramientas para lanzarse a la aventura de hacer pensar a la máquina. Solo queda por resaltar que en cada sección se debe manejar la parte gráfica colocando los tableros correspondientes.

Este método de separar por bloque cada función nos facilitará la tarea ya que con él se hace una disección del programa, lo que a su vez facilita su construcción.

Así terminamos de exponer las condiciones con que se puede construir el programa y estamos capacitados para dar un nuevo paso en el camino de los juegos de estrategia.

Fernando Pedró

DEFENDER OF THE CROWN

Para aquellos que quieren que los juegos de video sean algo más que una matanza de extraterrestres o una carrera de obstáculos, "Defender of the Crown" llega para lanzar un desafío: luchar por la corona y unificar el reino bajo el mando de un sajón.



Corre el año 1199; el rey retorna de las cruzadas y hace caballeros a seis grandes guerreros que lo llevaron a la victoria sobre los infieles. Una multitud de súbditos corre al palacio para ver la ceremonia.
Los nuevos caballeros se retiran y, en el camino a nuestra nueva posesión, ya que tomamos la personalidad de uno de ellos para este juego, atravesamos el bosque de Sherwood. Alli nuestro amigo Robin de Locksley nos da las malas noticias: el rey fue asesinado, la corona desapareció y se aproxima la guerra civil, ya que sajones y normandos se acusan mutuamente del crimen.
Esta es la historia tal como es contada tras la presentación, estilo cinematográfica, del juego. Desde el principio comenzamos a ver gráficos espectaculares, efectos sonoros y melodias que nos transportan al medioevo. El manejo se realiza totalmente mediante un menú de opciones y el joystick. Desde nuestra primera posesión, un castillo, debemos comenzar la conquista de territorios, uno por turno de juego. A la vez nuestros adversarios y aliados, a quienes maneja la computadora, van haciendo lo mismo. La conquista se realiza con el ejército, al cual tenemos que transferir hombres de la guarnición. Durante todo el desarollo del juego debemos velar por dejar compensadas nuestras fuerzas para que los ataques rivales no prosperen (si perdemos el castillo original, finaliza el juego). Por turno recibimos un ingreso mensual, en metálico, de acuerdo con los territorios que poseamos.

LAS OPCIONES DEL JUEGO

Torneo: Justa de caballeros armados con lanzas. Gracias a los excelentes efectos gráficos, en el momento del choque (vemos a nuestro adversario venir galopando) se produce un efecto bastante real. La contienda puede ser por fama o por algún territorio; puntualmente es la parte más dificil del juego, por lo menos nosotros no conocemos a nadie que haya salido vencedor de ella, por lo que recomendamos que compitan solamente por la fama hasta que le encuentren la vuelta.
Conquista: Consta de tres partes: leer el mapa, mover la tropa y transferir elementos. Con la primera podemos ver quién ocupa un territorio, cuántos vasallos tiene y cuánto dinero recibe por mes por poseerlo. La transferencia solamente la podemos hacer en un castillo que sea nuestro, de la guarnición al ejército o viceversa. La movilización de tropas es la conquista en sí. Si hay demasiada diferencia entre ambos ejércitos, se produce la invasión directamente, de lo contrario hay un combate en que debemos movernos a través de un submenú para adaptar la estrategia a seguir según el balance de fuerzas. Las opciones son: ataque feroz, lucha frontal, ataque de flanco, bombardeo y retirada (por si nos llega a ir mal). Si los atacados somos nosotros, se produce una situación similar.

Ataque por sorpresa: Para aventureros. Esta opción nos permite

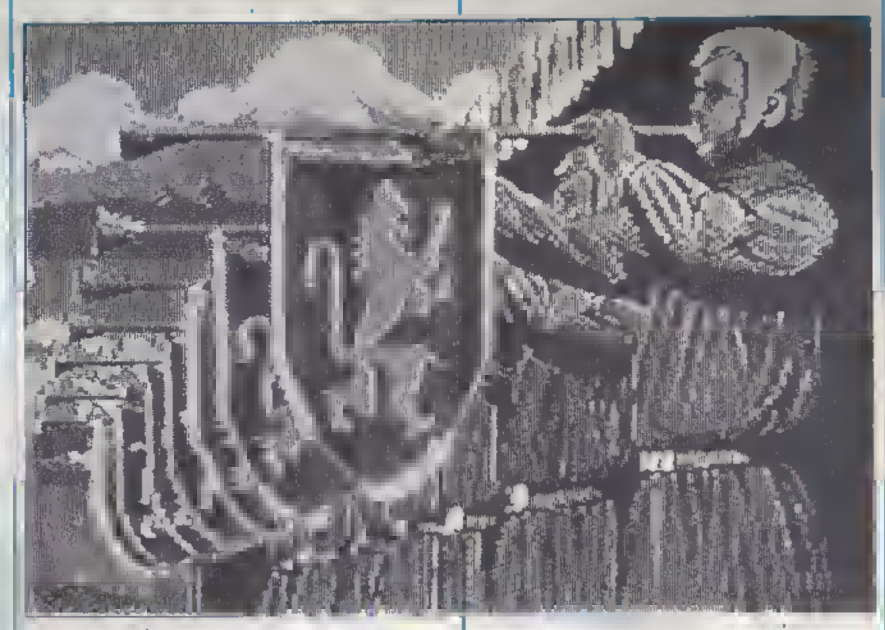
hacer incursiones nocturnas en castillos y, si la suerte nos acompaña, conquistarlos. Claro que, después, lo difícil es mantener esa posición, por lo que solamente es recomendable hacerlo si el castillo está rodeado completamente por nuestras tropas (de esta manera no damos posibilidad para el contrataque).

Comprar tropa: Con el metálico acumulado podemos, utilizando nuestro turno, comprar soldados, caballeros, catapultas o castillos. El precio de los mismos es 1,8 ,15 y 20 respectivamente.

Leer mapa: Al igual que la opción de conquista, esta nos permite acceder a información sobre los diversos territorios desde el menú principal.

Pasar: Se deja pasar el turno sin hacer nada más.

El juego viene en un disco grabado en las dos caras. La primera esta íntegramente cubierta por la presentación y la elección, por parte nuestra, del personaje: Wilfred de Ivanhoe, Wolfric the Wild, Geoffrey Longsword o Cedric de Rotherwood. Luego se da vuelta el disco y comienza el juego. Las alternativas son variadas. Por ejemplo, podemos ir a rescatar a una princesa raptada por un normando, o podemos sufrir alguna pérdida en metálico, en tropas o en territorio por las sucias artimañas de nuestros enemigos, que confabulan a nuestras espaldas.



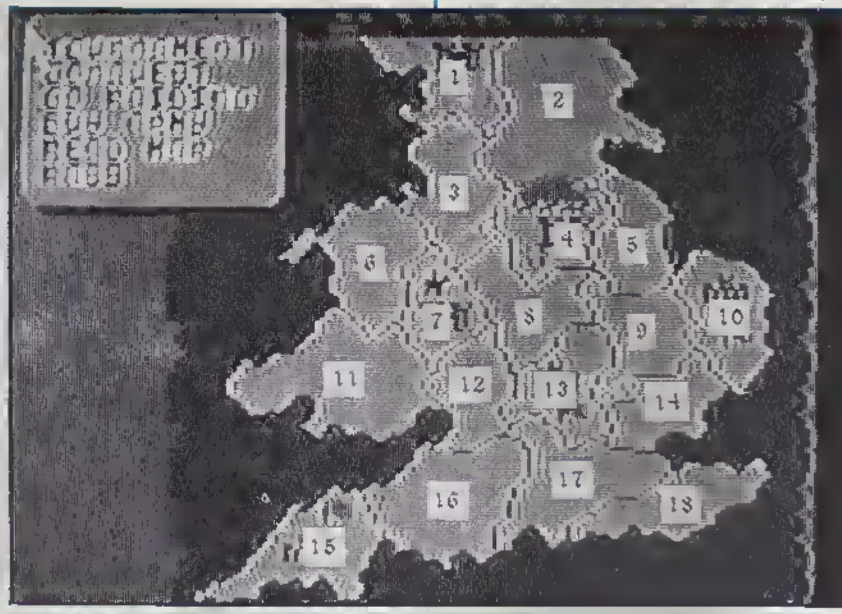
UNA ESTRATEGIA

Realmente fue muy duro, pero con-

seguimos coronarnos luego de conquistar todo el territorio. La corona aparece y el "ladrón" es revelado sobre el final, antes del cartelito "The end". El juego comienza en octubre de 1199 y estuvimos jugando mucho tiempo sin poder sobrevivir del año 1201. Finalmente, cuando logramos el objetivo, la batalla decisiva se produjo en abril de 1204. Pero, ¿cuál fue la estrategia que nos llevó a esa victoria? Primero y fundamental, no luchar contra nuestros aliados sajones (los de los escuditos) al comienzo del juego, aunque ellos nos ataquen. Hay que buscar paso libre a través de sus tierras (cosa que se nos ofrece) para ir a luchar contra los normandos. Debemos buscar regiones distintas de las que están atacando ellos, para poder concentrar tropas y no dejar que ningún normando se expanda demasiado. Además es importante para dejar comunicadas a



nuestras tropas y que no sea fácil atacarnos. Por ejemplo, conquistando el condado de Sussex, en la punta inferior derecha de la isla, nos aseguramos un ingreso mensual de 7 monedas, y al mismo tiempo es fácil de defender ya que solamente puede ser atacado desde Hampshire. En cambio Leicester, en el centro del mapa, puede ser atacado desde 7 territorios y nos ofrece solo 2 monedas mensuales. Cuando tengamos demasiados territorios y una buena cantidad de oro en nuestras arcas, habrá que empezar a fortificarlos, construyendo castillos, dejando una prudencial guardia en los mismos para su seguridad. La compra de armamento debe ser periódica, para evitar los saqueos que reducen nuestras riquezas a la mitad. Nosotros compramos cada tres o cuatro meses según la situación coyuntural. El ataque a los castillos lo dejaremos para el final, cuando se tenga una a-



TERRITORIO	VASALLOS	INGRESO POR MES
1 CUMBRIA	0	3
2 YORKSHIRE	6	2
3 LANCSHIRE	6	4
4 NOTINGHAM	0	2
5 LINCOLNSHIRE	6	3
6 GHYNEDD	4	2
7 CLHYD	0	3
8 LEICESTER	5	2
9 CAMBRIDGE	6	5
10 NORFOLK	0	6
11 GLAMORGAN	4	1
12 GLOUCESTER	12	7
13 BUCKINGHAM	0	8
14 ESSEX	8	6
15 CORNWALL	0	8
16 DORSET	6	4
17 HAMPSHIRE	8	3
18 SUSSEX	3	7

brumadora diferencia numérica. Estos ataques constan de dos partes. Un sitio de seis días, durante el cual lanzamos con una catapulta piedras y tizones encendidos para diezmar a la guarnición, y luego la batalla, en la que debemos tener una buena ventaja material para poder vencer. En el ataque con catapultas recomendamos primero las piedras para romper el muro y poder lanzar por allí nuestras "bolas de fuego" en los últimos días del sitio. Con esta estrategia, y una buena dosis de suerte (¿qué vencedor no la tiene?), conseguimos el objetivo luego de un par de horas de juego. Pero previamente hubo muchísimos intentos fracasados, así que no hay que desanimarse si los primeros resultados son malos. Creemos también que las posibilidades tácticas no se acaban aquí, ya que se puede investigar qué sucede con una política de alianzas (casándonos con la princesa), logrando vencer en los torneos o cómo aprovechar los ataques por sorpresa. Este entretenimiento, de origen inglés, está llamado a convertirse en uno de los favoritos de los "juegologos", ya que ofrece una simulación de juego de guerra que nos hace recordar la aparición del **TEG** entre los juegos de tablero.

FILTROS

Autor: Alfredo H Termeniello
Comp.: DC 64/C
Clase: Educativo/Musical



Muchas personas creen que hacer un programa de música es muy difícil, o que hay que saber mucho sobre música para poder hacerlo.
Nuestra intención es "desmitificar" esto. Para ello publicamos aquí este programa que nos enceñará muchos sonidos que podremos sacarle a nuestra computadora.
El autor del mismo, **Alfredo Termeniello**, es un entusiasta colaborador de nuestra revista. El no sabe música (sólo puntea con su fiel guitarra las melodías) pero encontró en la Drean Commodore 64 un instrumento con el cual canalizar su sentido musical. Se encontró con muchos inconvenientes para lograr sacarle las notas a la máquina pero, como nos confesó, ello se debió más que nada a que no contaba con un material didáctico apropiado.
Por esa razón, para ayudar a quienes, como él, quieren usar a la computadora para hacer sus melodías o reproducir la de los demás, hizo una serie de programas que estamos publicando. En ellos muestra su técnica y distintas aplicaciones de la misma.
El presente programa, hecho con los fines educativos mencionados, nos enseña a utilizar los filtros de la Drean Commodore 64/C.
Podemos ver en el cuadro, por ejemplo, cómo utilizar el filtro presente en la posición 54295 para filtrar una, dos o las tres voces.
Una vez cargado el programa, basta con darle "run" para que comencemos a escuchar una escala de sonidos. En pantalla aparecerán los números de las diferentes posiciones de memoria. Cuando hayamos escuchado uno que nos guste, nos bastará con apretar "stop" y anotar cuales son esos registros.
Luego podremos utilizarlos en nuestros programas. Para continuar la ejecución del programa que se detuvo, colocamos "cont" y este se ejecutará desde donde fue parado.
Las aplicaciones de este programa van desde las educativas (mencionadas anteriormente), hasta de ayuda al programador que desee colocarle música y sonidos a sus programas.
En los próximos números continuaremos con estas notas, ofreciendo aplicaciones sobre programas con melodías elaboradas en los que interviene más de una voz.

Cuadro 1

```
..E 54295,01:     FILTRO VOZ 1
..KE 54295,02:    FILTRO VOZ 2
.. E 54295,04:    FILTRO VOZ 3
.. E 54295,05:FILTRO VOZ 1 Y VOZ 2
.. E 54 95,06:FILTRO VOZ 1 Y VOZ 3
.. E 54295,0 : ..TRO VOZ 2 Y VOZ 3
.. ..E.. ..... .RO VOZ 1-2 Y 3

***************************************
POKE 54296,01-15:   VOLUMEN.
POKE 54296,16+VOLUMEN:FILTRO PASABAJO
POKE 54296,32+VOLUMEN:FILTRO PASABANDA
POKE 54296,64+VOLUMEN:FILTRO PASA..
POKE 54296,112+VOLUMEN:FILTRO ...
```

```
 ************** .      ****************
 ****     ..     ..              ****
4 ***   ..                       ***
5 ****          .  S             ****
6 ***        CALLE 76 NRO. 41     ***
7 ****          BERISSO          ****
8 ***       LA PLATA (BS.AS.      ***
9 ************************************
10 POKE 53280,.:POKE 53281,.
20 X$="  PULSE RETURN   ":D$="DIENTE DE SIERR.
30 FILTRO=16:VOLUMEN=15:H=32:GOTO 80
40 FILTRO=16:VOLUMEN=15:H=128
5. D$="RUIDO":RESTORE
70 X$="  PULSE RETURN   "
80 FOR D=54272 TO 54296:POKE D,0:NEXT
90 POKE 54293,1 . :POKE 54294,1
100 POKE 54295,241 :POKE 54296,31
110 POKE 54277,11  :POKE 54278,11
120 PRINT"  ":READ A$
130 FOR C=1 TO 155 STEP 8:POKE 54276,H+1
140 POKE 54272,48 :POKE 54273,4
150 PRINT"  FILTROS":PRINT"  POKE 54293, "INT(
   /32)"  POKE 54294, C
160 IF FILTRO>.80 THEN FILTRO=112:
170 PRINT"  POKE 54295, 241: POKE 54296, FILTRO+
   VOLUMEN
   POKE 54296,FILTRO+VOLUMEN:PRINT"
   PRINT" NOTA
   PRINT" POKE 54273, 48 :POKE 54273, 4":POKE
 5427.,H
   PRINT"
   PRINT"  TIPO DE ONDA       ";"D$
   PRINT"                   =POKE 54276,"H+1
   PRINT"
   PRINT"  TIPO DE FILTRO  ";A$;"=FILTRO
260 PRINT"             =POKE 54..        RO"      UMEN"
270 PRINT"                              ":FOR
   D=     2.        :GOSUB
         54293,                 POKE 542
       ":PRINT:PRINT
   FILTRO=FILTRO+FI .
31.      1. .0    128 AND H=12
3..   IF FILTRO    128 THEN 350
330 PRINT"           "X$
340 PRINT"GOTO 90           ":END
350 PRINT"            "X$
360 PRINT"RUN  40          END
370 PRINT"            "X$
380 PRINT"  RUN 390             END
390 PRINT"
400 PRINT"  POKE 54295,241-         (C) BY
410 PRINT"  RESONANCIA     (0-240)      BARBATO
420 PRINT"  VOZ A FILTRAR  (1-15)       1987.
425 PRINT"
430 PRINT" 240=MAXIMA RESONANCIA-0 SIN RESONANCIA
440 PRINT"   1=VOZ 1 FILTRADA  ":PRINT"    -":PRINT"
  -241"
450 POKE 54277,31 :POKE 54278,15
460 POKE 54296,31 :POKE 54275,33
470 FOR K=0 TO 170 STEP 4
480 POKE 54272,40 :POKE 54273,4
490 POKE 54294,K  :NEXT
500 GET S$:IF S$=CHR$(13) THEN RUN
510 POKE 54276,33 :FOR K=170 TO 0 STEP-4
520 POKE 54272,48 :POKE 54273,4
530 POKE 54294,K  :POKE 54276,32:NEXT
540 POKE 54276,32 :GOTO 450
550 FOR D=1 TO 5
560 POKE 781,10:POKE 782,17:SYS65520:PRINT
570 POKE 781,10:POKE 782,17:SYS65520
580 PRINT A$:NEXT
590 RETURN
600 DATA"  PASA-BAJO  ","  PASA-BANDA  ","
   PASA-ALTO  ","  *** TODOS ***  "
```

BASIC TORTUGA (II)

Mostramos las ventajas del C como lenguaje estructurado y cuál es la mejor forma de acercarse al mismo.

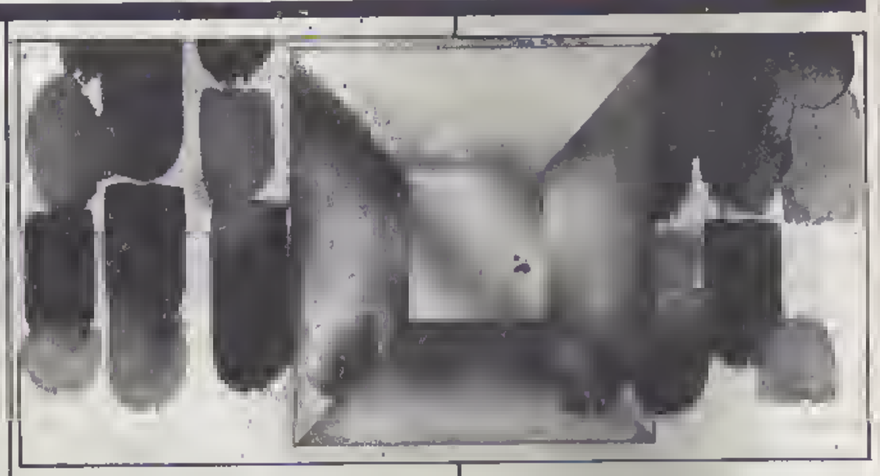
Como es costumbre en muchos manuales de C, comenzamos la parte práctica del lenguaje saludando al mundo a través de este súper-conocido programa:

```
main()
[
    printf("hola a todos");
]
```

Lo que se ha visto arriba solamente imprime la reconocida leyenda en castellano a pesar de contar con dos vocablos alienígenas. Es el equivalente en C del programa en BASIC:PRINT "hola a todos"

Un programa en C es una gran función que para realizarse necesita la ayuda de otras funciones accesorias. La gran función se llama Main, y todo programa debe contar con ella. Cuando corremos nuestro programa, empezamos el flujo del recorrido por allí. En su derrotero, seguramente se encontrará con otras funciones y acudirá a ellas. Las funciones accesorias pueden estar programadas explícitamente en el programa o bien solamente declaradas, en cuyo caso se sobreentiende , que pertenecen a alguna librería. En nuestro caso no hay ninguna función accesoria explícitamente programada salvo Main, el programa propiamente dicho.

Lo que sí hay es una función que supuestamente está en una librería. Printf es una función "de alto nivel"; no es propia de C. Dependería, en un primer momento, del compilador. Pero se ha difundido tanto que hasta los mismos autores del lenguaje las adoptaron casi como estándar, así que podemos considerarla como parte del lenguaje. De todas maneras, las burdas consideraciones humanas son poco aceptadas por las computadoras, y cada vez que el compilador se encuentre con Printf, irá a la librería porque no reconocerá la palabra como propia. No viene al caso explayarse acerca de la manera de declarar una función de librería. Es más, a lo largo de esta nota se omitirán muchas formalidades que serán presentadas el día que se compren su propio compilador C y algún libro sobre el tema. La idea es entusiasmarlos y abrirles la mente a cosas nuevas. Lo que puedo decirles es que en la mayoría de los compiladores se pueden mezclar listados de diversos programas como en un procesador de textos, y de esta manera incluir en el archivo del programa secciones declaratorias que prácticamente deben estar en toda cosa que uno programe. Estos archivos suelen tener tipeados los protocolos de declaración de funciones externas (de librería), y lo que uno tiene que hacer es solamente aclarar en el listado que en el momento de la compilación se deberá pegar el archivo que tiene las declaraciones de las funciones a usar. No importa si una función está declarada y no se usa nunca; eso no afecta al desenvolvimiento del programa. Es por eso que la mayoría de los "encabezadores de programas" incluyen sin preocuparse casi todas las funciones que el compilador implementa.



Repasando un poquito, cuando el compilador se encuentre con la declaración de la función (ausente en el listado de arriba) será advertido de que cuando halle, en el listado esa función tendrá que ir corriendo a la librería y ver de qué se trata. Una vez que se terminaron todas las declaraciones (y eventualmente funciones y procedimientos), llegamos a Main. En esta sección, el compilador comienza a trabajar. Lee printf, reconoce su ignorancia, corre a la librería para ver qué es y vuelve con las novedades. Cuando da por terminada la compilación, y si es un tipo "piola", llamará al "linkeador" para terminar de armar el programa.

Recalco lo de "piola", porque algunos compiladores suelen, al fin de la compilación, sacarles la lengua a los usuarios y decirles "...ahora linkeás vos, vago..."

LA IMPORTANCIA DE LOS PARENTESIS

Como habrán notado, en el primer ejemplo hay dos palabras del lenguaje y 9 signos. Como toda función debe estar preparada para recibir parámetros, los paréntesis no pueden omitirse nunca. Pero no siempre es obligatorio cargar una función con un parámetro (como anticipo, un procedimiento -en líneas generales- es una función que no devuelve valor alguno, y que muchas veces no

necesita ningún parámetro). Main será el representante del programa, el director de orquesta de la sinfónica C; él es el que debe dar las órdenes a las funciones con los valores adecuados y, como buen dirigente, no debe ser mandado por nadie. Y digo en líneas generales porque casi siempre, y no siempre, Main se las arreglará solito. Para nosotros la función principal será absoluta e imperial: nadie le impondrá nunca nada. De todas maneras, sacar los paréntesis de los parámetros sería demasiada simplificación y por eso quedan.
Las llaves que siguen a Main y cierran el programa son los medios por los cuales uno puede separar conjuntos de sentencias. En este caso sólo sirven para delimitar el programa, pero más adelante veremos su uso.
Finalmente llegamos a printf: una función completa, con parámetros y todo. Lo que está encerrado entre paréntesis en esta función será impreso de la manera usual. El punto y coma final delimita una sentencia; es la manera por la cual el compilador detecta el "fin de línea". Este es otro programa similar:

```
main()
[
   printf("hola a todos");
   printf("espero que me aprendan");
   printf("pronto");
]
```

Y no tiene gran ciencia; sólo 3 printf separados por puntos y comas. El siguiente ejemplo típico en los manuales de C es el de la tabla de conversión de grados centígrados a farenheit:

```
main()
[
   int tempinferior,tempsuperior,step;
   float far,celsius;

   tempinferior=0;
   tempsuperior=300;
   step=20;

   fahr=tempinferior;
   while (far<=tempsuperior)
   [
      celsius=(5/9)*(far-32);
      printf("ccaaddeennaa",far,celsius);
      far=far+step;
   ]
]
```

Ajá...Pero no importa...Uno después entiende.

Las palabras reservadas int y float encabezan la declaración de variables enteras y de punto flotante (decimales) respectivamente. Las variables que están después de int serán enteras; las que estén después de float, decimales.

Les siguen tres sentencias de asignación que tiene el mismo sentido que en BASIC.

Y acá hay que agarrarse. La estructura general de un while es la siguiente:

```
while (pase tal cosa)
[
   se deberá hacer esto
]
```

La condición del while va dentro de los paréntesis. Mientras sea verdadera, se realizará un ciclo pasando por las instrucciones que se encuentran dentro de las llaves debajo del while. El ciclo asignará el valor a celsius, lo imprimirá junto al valor de far correspondiente, y finalmente se incrementa el valor de far y evaluará la condición escrita en los paréntesis. Si se mantiene verdadera, sigue el próximo ciclo. Si no, sale a lo que continúa después de la llave de cierre del while.
La sentencia while puede ser reemplazada por otra que también reconoce el compilador:
for (asignación de comienzo; condi-

ción para mantenerse dentro del ciclo; incremento de alguna variable);

```
[
    hacer esto
[
```

Que, aplicado a la práctica, sería:

```
for(a=1;a<10;a=a+1);
[
    hacer esto;
]
```

Con su equivalente en BASIC: For a=1 to 10 step 1: hacer esto:next. Nos encontramos de nuevo solamente con Main. Acá tampoco hay funciones declaradas en un programa explícito. La diferencia entre la sentencia de impresión actual y la anterior radica en los parámetros pasados. Lo que hará printf será imprimir el valor de far y celsius de una manera especial. Esa manera especial viene dada por la "ccaaddeennaa" o cadena de control, que indica cómo deberán ser impresos los parámetros que figuran a continuación (que pueden ser la cantidad que uno quiera; pero ojo, es prácticamente la única función con número de parámetros variables). La notación usada es engorrosa pero muy dúctil; a título de comentario podemos aclarar que lo que iría en esa "ccaaddeennaa" es: %4.Of%6.1f/n. Y no es una cargada: cada uno de esos signos significa algo. Si se queda con la duda, vaya, compre un compilador C y un libro al respecto.

DEFINAMOS NUESTRAS FUNCIONES

Hasta ahora no habíamos dicho nada acerca de las funciones definidas por el usuario. Esta es la parte más interesante de la historia. Para una pasada superficial, es lo que define la diferencia entre BASIC y un lenguaje estructurado. Un buen ejemplo es la potenciación:

```
int power(x,n)
int x,n;
[
    int i,p;
    p=1;
    for(i=p;i<=n;i=i+1
    [
        p=p*x;
    ]
    return(p);
]
```

Acá tenemos un ejemplo de una función a la cual hay que pasarle los parámetros x y n. Su nombre es power y su parámetros son "x", un número entero, y "n", otro número entero, cuya declaración se ubica después del nombre de la función. La palabra int antes de power señala que la misma devolverá un número entero.

A continuación se abre una llave para indicar el comienzo de una función, e inmediatamente después aparece una nueva declaración. Las variables i y p serán números enteros locales a la función por estar dentro de las llaves. Existirán mientras la función sea invocada y cuando ésta devuelva el resultado se destruirán. Son variables auxiliares.

Posteriormente vemos una aplicación del ciclo for que es equivalente a:

p=1
for i=p to n:p=p*x:next

Y por último se devuelve el valor correspondiente a la rutina que llamó esta función. Return (variable) es la puerta de salida para una función. Cuando lleva una variable, la función devuelve ese valor; si aparece return() (sin nada dentro de los paréntesis), casi seguramente se tratará de un procedimiento (y hacemos la misma salvedad del "casi siempre" hecha antes). Esta sentencia no necesariamente debe estar al final de la función. Cuando el fluir de la rutina se encuentre con ella, volverá a la función inmediatamente anterior en orden de llamada, con valor requerido.

Recién dijimos que un procedimiento no devuelve valores. Esto no es del todo cierto pero puede servirnos como aproximación. Un buen ejemplo de un procedimiento que no devuelve valores son las rutinas que dibujan pantallas de presentación. La siguiente instrucción ejemplifica ese concepto:

```
void pantalla(caracter)
char caracter;
[
    int a;
    a=1;
    while(a<40)
    [
        putchar(caracter)
        a=a+1;
    ]
]
```

Este procedimiento no devuelve ningún valor (void, al comienzo, aclara esta característica). Recibe como parámetro un carácter, declarado con el tipo char. Después 40 veces,incrementa la variable dentro del ciclo while imprimiendo en cada incremento el carácter con el que invocamos el procedimiento, en fila. Veamos un ejemplo algo más integrado. El siguiente programa, usando las rutinas anteriormente descriptas, genera una pantalla de presentación compuesta por el título y un subrayado de 40 carateres y la potencia cuarta de seis:

```
main()
[
    int a;
    printf("Esta es una presentacion");
    pantalla('-');
    a=power(6,4);
    printf("Seis a la cuarta vale");
    printf("ccaddeennaa",a);
]
```

La "ccaaddeennaa" del printf debe indicar que lo que se manda a imprimir es un número entero. Nótese la diferente manera de usar las funciones y procedimientos. Un procedimiento como pantalla o printf se invoca como una instrucción común

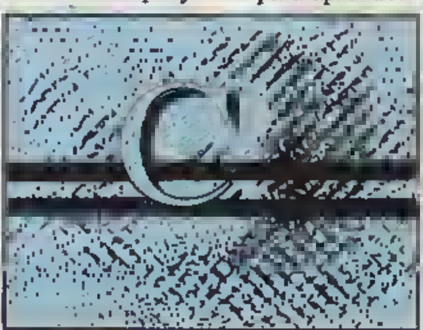
de BASIC con parámetros o sin ellos. Pero una función que devuelve un valor debe ser asignada a una variable; en líneas generales no puede aparecer suelta.

¿COMO APRENDER C?

Lo que hemos visto hasta ahora es sólo una infima parte del potencial de C. Ante todo, preferi mostrar las ventajas de C como lenguaje estructurado por encima de lo que es en la realidad. Tampoco tiene mucho sentido hacer un curso de C por revista; lo importante de estas notas es encender la curiosidad del usuario y hacer que apenas termine de leer la nota, vaya volando a la casa de soft más cercana a comprar un compilador C y un buen libro al respecto.

Como bibliografía para aprender, es muy recomendable **El lenguaje de programación C**, de Kernighan y Ritchie, que, por ser los autores de este lenguaje, convierten al libro en "oficial". Utilizan, un lenguaje muy ameno y directo, sin "verso" y con muy buenos ejemplos. Sin embargo hay que tener cuidado; es un lenguaje que conceptualmente no tiene nada que ver con el nivel del BASIC; no es recomendable para iniciar se en la programación.

El método que yo usé para aprender



este lenguaje fue el siguiente. Miré el libro a vuelo de pájaro, prestando atención a los programas de demostración y en especial al "orden" de los listados. Al mismo tiempo hice algunas copias del disco por si hacía alguna macana compilando. Y me dediqué a mirar archivos, la biblioteca (si es accesible), a ver cómo se manejaban los comandos del compilador, del editor y del linkeador, compilé lo programas de demostración, etcétera. Después de una semana de sondeo, abrí el libro en la primera página y lo leí con la computadora al lado. No se me escapó ningún ejemplo sin probar, colgué la máquina muchas veces, modifiqué muchos programas y, sobre todo, razoné los programas primero en lápiz (o birome) y papel. En menos de un mes de práctica fuerte había dominado gran parte de la plástica lingüística; y me largué a hacer las primeras cosas en serio.

Los programas que puse en esta nota, tienen una complejidad de una semana de aprendizaje (algunas horas por día). Con dos o tres meses más pueden imaginarse las cosas que pueden salir. Como punto de referencia, lo descripto en esta nota representa sólo un diez por ciento de lo que se puede hacer en C.

Alejandro Parise

PRIMER CAMPEONATO DE COMPUGOLF

Terminó el Primer Campeonato de Compugolf para computadoras Drean Commodore. Público de todas las edades mostró mucho entusiasmo ante la nueva experiencia y un nivel competitivo excelente. La competencia se dividió en amateurs y novicios y dada las expectativas despertadas quedó abierta la factibilidad de un segundo encuentro.

Con gran éxito se llevó a cabo el día 26 de septiembre el **Primer Campeonato de Compugolf Drean Commodore** en las instalaciones de la sede central del Centro de Atención al Usuario Drean Commodore.
El torneo estuvo abierto a la participación de compugolfistas de todas las edades, y en él se observó un gran entusiasmo y afán de victoria.



El nivel competitivo fue excelente y las alternativas del score fueron cambiantes, con varios participantes con chances al título hasta los últimos hoyos.

Próximamente se darán a conocer los handicap de la ADC (Asociación de Compugolf).

El torneo fue jugado con el programa **World Class Leader Board**,

CENTROS DE ATENCION AL USUARIO

Complementando la información brindada en nuestro último número, damos aquí la lista de los nuevos Centros de Atención al Usuario en distintos puntos del país.

CAPITAL FEDERAL

Centro de Atención al Usuario Pueyrredón 860, piso 9 (Sede Central).
Belgrano: V. de Obligado 2833
Caballito: Juan B. Alberdi 1196

GRAN BUENOS AIRES

Avellaneda: Av. Mitre 1802
Lomas de Zamora: Acevedo 48
Quilmes: Moreno 609
Ramos Mejía: Bartolomé Mitre 180

PROVINCIA DE BUENOS AIRES

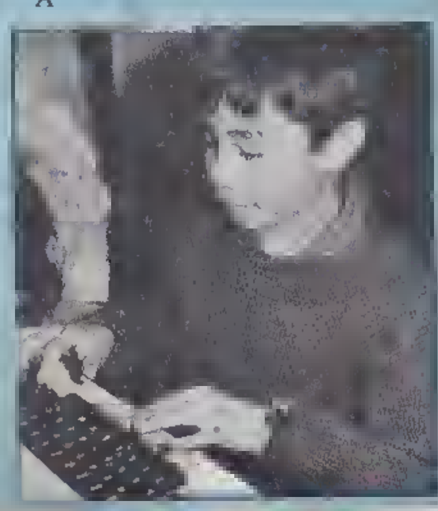
Bahía Blanca: Mitre 163/69
La Plata: Calle 50 número 637
Pergamino: Alem 532
Tandil: Rodriguez 769

PROVINCIA DE CORDOBA

Río Cuarto: Vélez Sarsfield 62

PROVINCIA DE CORRIENTES

Corrientes: Junín 1327, primer piso "A"



PROVINCIA DE ENTRE RIOS

Concordia: Urdinarrain 50

PROVINCIA DE LA PAMPA

General Pico: Calle 24 número 433

PROVINCIA DE MENDOZA

Mendoza: San Martín 1052, 5º piso "21"

PROVINCIA DE SANTA FE

Santa Fe: 4 de Enero 2770

PROVINCIA DE SANTA CRUZ

Río Gallegos: San Martín 1021

PROVINCIA DE TUCUMAN

San Miguel de Tucumán: San Juan 451

CENTRO DE ATENCION AL USUARIO



Los vencedores y sus trofeos.



La entrega de premios.

RESULTADOS DEL TORNEO

Jugado a 36 hoyos en las canchas de Saint Andrews y Gauntlet Country Club

CATEGORIA AMATEURS

Pos	Nombre	Golpes
1	Ramiro Pettina Goobar	132 (64-68) -12
2	Pablo Kawaguchi	141 (66-75) -3
3	Guido Konig	153 (74-79) +9
4	Rafael Rocchia	159 (79-80) +15
5	Pablo Konig	174 (85-89) +30

CATEGORIA NOVICIOS

Pos	Nombre	Golpes
1	Federico Qundjian	134 (67-67) -10
2	Jose Bello	136 (69-67) -8
3	Daniel Lorenzano	141 (73-68) -3
4	Diego Llanos	146 (74-72) +2
5	Pablo Vicente	148 (71-77) +4



La competencia en su desarrollo.

que comentaremos en nuestro próximo número.

Debido a la gran cantidad de participantes fue dividido en dos categorías: novicios y amateur, resultando ganadores **Federico Qundjian** y **Ramiro Pettina Goobar**.

En cuadro aparte ofrecemos las principales posiciones de cada categoría.

Se disputaron 36 hoyos en las canchas de Saint Andrews y Gauntlet Country Club, opciones C y D respectivamente del programa de golf mencionado, la última versión de la exitosa serie Leader Board.

Mención especial merecen **Sergio Freue y Sebastián Villafañe**, quienes lograron un hoyo en uno en el par 3 del hoyo 2 de la cancha Gauntlet Country Club, de 160 yardas.

Curiosamente ninguno de los dos consiguió figurar entre los integrantes del podio. Sin embargo nadie les quitará lo bailado (o embocado).

Ya se prevé la organización del próximo torneo a raíz del éxito del primero y del entusiasmo despertado.

LAPIZ OPTICO

A través de esta nota les explicamos el principio de funcionamiento de este interesante accesorio.

El lápiz óptico es, sin lugar a dudas, uno de los accesorios más atractivos que encontramos para ampliar las posibilidades de nuestra computadora.
Este diminuto elemento, del tamaño de un bolígrafo y que se conecta al port de joysticks, permite dibujar figuras sobre la pantalla de la misma forma en que lo haría un lápiz sobre un papel e incluso con algunas ventajas.

¿COMO DIBUJA?

Para comprender cómo es posible la representación en pantalla de lo trazado con el lápiz óptico debemos remitirnos a cómo la computadora genera la imagen sobre el monitor.
La imagen completa que nosotros percibimos está formada en realidad por un único punto luminoso que recorre la pantalla a través de sucesivas líneas horizontales desde el ángulo superior izquierdo hasta el ángulo superior derecho, como se observa en la figura 1.
Como este proceso se repite 50 veces por minuto el efecto resultante es una imagen completa. Esta ilusión se debe a la persistencia de la retina ocular y del fósforo de la pantalla.

Figura 1



Figura 2

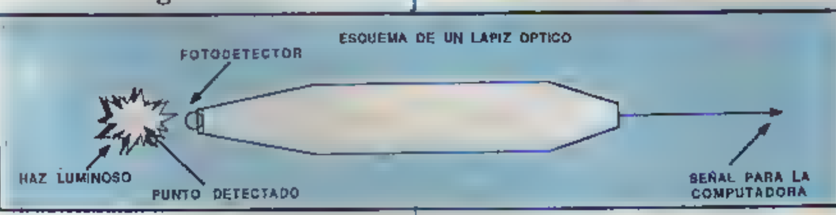
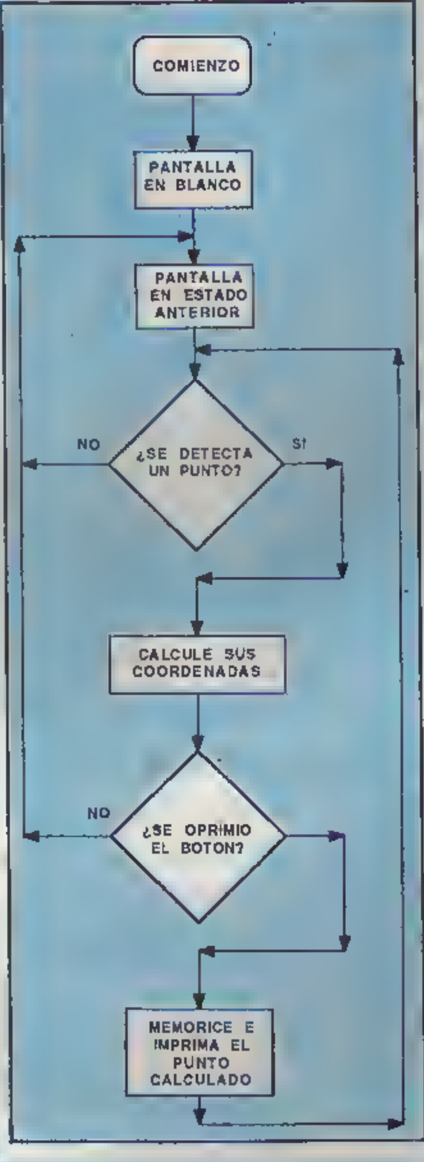


Figura 3



La operación de escritura del lápiz óptico se basa en un ojo electrónico que detecta el paso del punto luminoso por la pantalla. Al acercar el lápiz este recibe una luz y genera una señal que envía a la computadora (ver figura 2).

Internamente la computadora, que se encarga de generar el trazado, conoce en qué lugar exacto de la pantalla está el punto luminoso en el momento en que recibe esa señal del lápiz óptico.

Esta operación es realizada por el chip de video que dispone de dos registros para almacenar la posición del punto detectado. La posición horizontal (eje x) se almacena en el registro 19 y la posición vertical (eje y) se guarda en el registro 20.

Este punto es representado en la pantalla para que el dibujante pueda reconocer la ubicación del mismo.
Hasta aquí el funcionamiento exclusivo del dispositivo. Todas las posibilidades que este accesorio posea (dibujar, colorear, generar figuras geométricas) son atributos conferidos por el software utilizado.
Así, por ejemplo, se dispone de un botón que permite fijar los puntos dibujados. De lo contrario siempre tendríamos en la pantalla un único punto: el último.
En la figura 3 observamos un diagrama que facilita la comprensión de todo el proceso. En un principio la computadora genera una pantalla en blanco y pregunta si algún punto ha sido detectado por el lápiz óptico.
En el caso afirmativo almacena las coordenadas del mismo y pregunta si el botón está presionado. Si lo está, ordena que ese punto no sea borrado.
Cuando iniciamos el siguiente ciclo la computadora genera la pantalla con un punto e interroga si ha recibido algún punto nuevo.
Repitiendo este ciclo en forma indefinida se obtienen líneas completas sobre la pantalla.
Como vemos no es el lápiz quien dibuja sobre la pantalla, sino que la computadora representa sobre el monitor los puntos luminosos detectados por el lápiz.
En el mercado existen distintos programas destinados a sacarle el jugo a este accesorio permitiendo cambios de color, de trazo y generación de figuras.

TRUCOS

EL BUFFER DEL TECLADO

El buffer del teclado es como una pequeña zona de memoria en donde se van depositando los caracteres que se presionan en el teclado.

Gracias a él se debe el efecto de que sigan apareciendo letras cuando nosotros no estamos escribiendo.

En la Drean Commodore 64/C existen varias técnicas que utilizan este buffer para generar órdenes en forma independiente.

Por ejemplo si nosotros tipeamos:

```
5 REM EJEMPLO DE
COMANDO AUTOMATICO
10 POKE 631,131
20 POKE 198,1
RUN
```

veremos como en la pantalla aparece el mensaje LOAD. La máquina supone que nosotros tipeamos textualmente ese comando.

Lo que hicimos fue ingresar el código de ese comando y luego decirle a la computadora, o mejor dicho al intérprete de comandos, cuantos caracteres habia en el buffer. Esta técnica también es utilizada para definir las teclas de función con comandos preestablecidos.

Para ello lo que se hace es variar el vector de interrupción para que chequee en forma constante si alguna de las teclas de función fue presionada.

Si es así se determina que comando se deberá ejecutar.

Para ello se toman los caracteres que lo forman (o su abreviatura) y se depositan en el buffer del teclado.

Luego se pone la cantidad de caracteres esperados y se salta a la rutina normal IRQ.

Esta verá que en el buffer hay algo, por lo que procede a tomarlos e imprimirlos.

Si nosotros como último caracter hemos puesto el código correspondiente al RETURN, entonces el comando se ejecutará.

DESACTIVA RESET

Los siguientes POKEs podrán utilizarlos no solo para desactivar las teclas RUN, RUN-RESTORE sino que, también, el reset interno de la C-128.

A través de estos dos POKEs evitaremos que por medio de las teclas de interrupción normales el programa deje la ejecución del programa actual.

En parte, este efecto se logra ya que se modifica la dirección de salto a la rutina RESTORE. Como sabemos, cada vez que nosotros presionamos dicha tecla se provoca una interrupción interna llamada NMI, que activa la ejecución de un programa interno que chequea por la presión de la tecla RUN.

En este método se pasa por encima ese chequeo, evitando que el programa pueda ser interrumpido.

```
10 REM DESACTIVA
RUN-RESTORE C-128
20 REM PARA DESACTIVAR RUN-RESTORE
25 REM TAMBIEN ANULA RESET INTERNO
26 :
30 POKE 800, 112
40 POKE 792, 98
```

DIRECTORIO INVISIBLE

Cuando hacemos un programa, sobre todo si está basado en una idea original, o utilizamos archivos que contienen información importante y secreta, es natural, por una cuestión de "conservación", que queramos que no haya espías que estén husmeando en el directorio del disco para ver qué es lo que tenemos allí almacenado.

Este programa que nos envió **Jorge Franco** protege a los discos haciendo que su directorio no pueda ser visualizado, y tiene la ventaja además de que èl proceso es reversible y con el mismo programa podemos volver a hacer visible el listado de programas y archivos.

Para que funcione solo hace falta tipearlo y seguir las instrucciones, colocando "I" si queremos invisibilidad y "V" para el proceso inverso. Después, si queremos ver el directorio, probemos colocar el habitual LOAD "$", 8 y LIST y veremos cómo la computadora responde con un lacónico READY.

```
100 REM --------------------
110 REM DIRECTORIO INVISIBLE
120 REM --------------------
130 PRINT CHR$(147)
140 PRINT"INSERTAR DISKETTE Y PULSAR RETURN'"
150 GETX$:IFX$<>CHR$(13) THEN150
160 PRINT"INVISIBLE O VISIBLE (I/V)?"
170 GETX$:IFX$<>"I"ANDX$<>"V"THEN170
180 OPEN15,8,15:OPEN2,8,2,"#"
190 IFX$="I"THEN390
200 REM ***DIRECTORIO VISIBLE***
210 PRINT#15,"U1";2;0;18;0
220 PRINT#15,"B-P";2;144
230 GET#2,X$
240 IFX$=CHR$(20)THEN270
250 PRINT"EL DIRECTORIO ES VISIBLE"
260 GOTO 630
270 PRINT#15,"B-P";2;200:DN$=""
280 FORI=1TO16
290 GET#2,X$
300 DN$=DN$+X$
310 NEXT
320 PRINT#15,"B-P";2;144
330 FORI=1TO16
340 PRINT#2,MID$(DN$,1,1);
350 NEXT
360 PRINT#15,"U2";2;0;18;0
370 PRINT#15,"ID"
380 GOTO630
390 REM***DIRECTORIO INVISIBLE***
400 PRINT#15,"U1";2;0;18;0+0
410 PRINT#15,"B-P";2;144:DN$=""
420 FORI=1TO16
430 GET#2,X$
440 DN$=DN$+X$
450 NEXT
460 IF LEFT$(DN$,1)<>CHR$(20)THEN490
470 PRINT"EL DIRECTORIO ES INVISIBLE"
480 GOTO630
490 PRINT#15,"B-P";2;200
500 FORI=1TO16
510 PRINT#2,MID$(DN$,I,1);
520 NEXT
530 DN$=CHR$(20)+CHR$(20)
540 FORI=1TO13
550 DN$=DN$+CHR$(0)
560 NEXT
570 PRINT#15,"B-P";2,144
580 FORI=1TO16
590 PRINT#2,MID$(DN$,I,1);
600 NEXT
610 PRINT#15,"U2";2;0;18;0
620 PRINT#15,"ID"
630 CLOSE2:CLOSE15
640 END
```

OPEN

"¡Abrete Sésamo!", y la roca se corre dejando pasar al interior de la cueva a los famosos ladrones. El Open es la palabra BASIC que, como en el famoso cuento oriental, nos abre la puerta a los "tesoros" que se encuentran en los dispositivos periféricos.

El sistema computadora básico está compuesto por la unidad central de proceso (CPU), que es el cerebro de la máquina, un teclado para introducir información y una pantalla de video por la cual el sistema responde.

Sin embargo lo más común es tener conectadas otras unidades físicas al sistema: casetera, disquetera, impresora y otras.

Para transmitir datos a una de estas unidades físicas o desde ella es necesario indicar a cuál se está llamando, ya a que cada una tiene un tipo específico de interfase y una forma particular en la que tiene que recibir su entrada (la salida de la CPU).

Para "hablar" o "escuchar" a otro dispositivo se necesita primero abrir un canal lógico para ese dispositivo. Esto se realiza mediante la sentencia OPEN. Con ella se designa qué canal se va a usar y qué dispositivo se quiere utilizar.

El formato general para una sentencia OPEN es el siguiente:.

OPEN nc,nd,ds,nombre del fichero

donde nc es el número de canal que se utiliza para acceder al fichero (puede ser cualquier número entre 0 y 255), nd es el número de dispositivo periférico que se está llamando, ds es la dirección secundaria que establece ciertos parámetros para el dispositivo y que se detallan en la tabla; y nombre del fichero se utiliza para saber de qué fichero estamos hablando en una operación de lectura o escritura. Vamos a ver ahora las diferentes variantes que admite el Open para los distintos periféricos.

PROGRAMA 1

```
10 OPEN 1.1.2. [illegible].TOS"
20 INPUT A$
30 IF A$="FIN [illegible] HEN 60
40 PRINT#1.A$
50 GOTO 20
60 CLOSE 1
```



UNIDAD DE CASETE

El método más ordinario de almacenar datos es imprimirlos, uno por uno, en la unidad de casete.

El siguiente programa escribe datos en un fichero hasta que le introducimos la palabra "FIN". La orden PRINT¿ se utiliza para transferir los datos a la unidad periférica. La computadora colocará automáticamente una marca de fin de fichero al cerrar el canal, ya que cuando se abre se utiliza la dirección secundaria 2 para ese fin.

(ver programa 1)

Para leer esos datos debemos abrir el canal con la dirección secundaria 0 (lectura) y luego utilizar dicho canal para la transferencia de datos.

(ver programa 2)

ST es una varible especial que indica el estado de los dispositivos externos. Si el valor de ST es 64, entonces se ha llegado al final del fichero.

UNIDAD DE DISQUETE

El formato del comando OPEN presenta una leve variación para cuando es usada la unidad de disquete:

OPEN nc, nd, ds, "ndr: nombre de fichero, SEQ,W"

PROGRAMA 2

```
10 OPEN 1,1,0,"DATOS"
20 INPUT#1,A$
30 PRINT A$
40 IF ST<>64 THEN 20
50 CLOSE 1
```

TABLA DE NUMEROS DE DISPOSITIVOS PERIFERICOS Y DIRECCIONES SECUNDARIAS

DISPOSITIVO	NUMERO DE DISPOSITIVO	DIRECCION SECUNDARIA	OPERACION REALIZADA
Teclado	0	ninguna	
Unidad de casete	1	0	apertura para lectura
		1	apertura para escritura
	2	2	apertura para escritura, pero añadir marca de fin de tabla (EOT) en el cierre
Pantalla de Video	3	ninguna	
Impresora	4 5	7	Conjunto de caracteres alternativos
Unidad de disquete	8	0	Cargar un fichero de programas en la computadora
	9	1	Conservar un fichero de programas desde la computadora
		15	Orden de apertura/estado del canal
Otros dispositivos	5, 6, 7, 9, 10, 11		Los números de dispositivos y las direcciones secundarias son seleccionados y asignados por el fabricante del dispositivo

donde ndr es el número de unidad de disquete (para quienes posean un sistema con más de una), SEQ indica que es un fichero de datos secuencial. También puede usarse REL (relativo), PRG (programa) o USR (usuario). La W (o R) indica modo de escritura (o lectura).
En la Commodore 128 se halla además el comando DOPEN, que abre directamente el canal de disco deseado. DOPEN¿nc,"nombre del fichero,(S/P)", Llr, Dndr, Und, w
S es secuencial, P programa L la longitud del registro. El comando se puede ejecutar una vez que se introdujo el nombre del fichero ya que asume a los otros valores como estándar.

IMPRESORA

La comunicación con la impresora se establece a través del nd=4 (OPEN 1,4). La dirección secundaria 7 se puede colocar si se desea utilizar el juego de caracteres alternativo (mayúsculas/minúsculas). Como en el caso de las unidades de casete o disquete podemos direccionar una salida a la impresora mediante la sentencia PRINT#, o cambiando la salida principal , que habitualmente es a la pantalla, a otro dispositivo. Esto se realiza con el comando CMD, cuyo formato es CMD nc. Por ejemplo:
OPEN 1, 4: CMD 1
dirige toda la salida a la impresora. Así podemos utilizar PRINT para escribir directamente sobre la hoja de impresión.

CLOSE

Después de terminar el acceso a un fichero, se le debe cerrar, ya que el número de canales que se puede tener abierto simultáneamente es limitado. El comando que cumple con esta función es el CLOSE, y se debe colocar:
CLOSE nc.
En la Commodore 128 se tiene además el comando DCLOSE para cerrar un canal de disco.

OTROS DISPOSITIVOS

La entrada y salida se realiza habitualmente desde el teclado y a la pantalla por lo que estos dispositivos tienen naturalmente abiertos canales de comunicación con la CPU. Sin embargo se puede utilizar el comando OPEN para aprovechar el INPUT# y hacer una entrada directamente desde el teclado sin que aparezca el símbolo "?" del INPUT.
Para finalizar, nos resta resaltar que la importancia de la sentencia OPEN y del aprendizaje de su uso radica en las posibilidades que nos ofrece para utilizar los dispositivos conectados a nuestra computadora: "nuestras riquezas". Abrete Sésamo ...

CONDICIONES MULTIPLES

Este trabajo participa en el Concurso "Notas, Trucos y Programas". Lo realizó Guillermo Roit en un Drean Commodore 64/C, y nos ayuda a hacer más cortos ciertos programas.

Si alguna vez copiaron un programa de BASIC tal vez se hayan topado con una condición múltiple, del tipo: IF ((PEEK(56320) AND 16)=0) OR (PEEK(203)<>64) THEN 200.

Al pasar por esta línea, el programa saltará a la línea 200 si se presiona una tecla, o si se aprieta el botón de un joystick conectado en el port número 2. Esta sentencias, si bien son prácticas, se tornan molestas cuando aparecen varias veces en un programa.

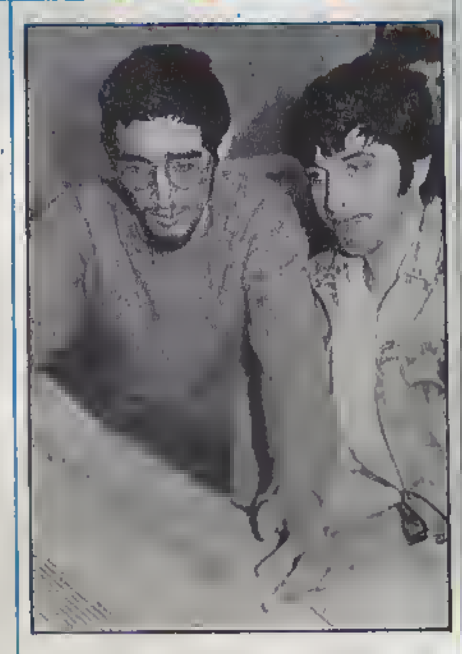
En otros casos, cuando una serie de pasos aparece en un programa en reiteradas ocasiones, se los pone en una subrutina, ahorrando así memoria y haciendo más claro y comprensible el programa. Sin embargo, en esta oportunidad la solución no es muy práctica.

La pregunta es: ¿será posible "sacar" estas condiciones múltiples fuera del programa principal como si fuera una subrutina? El BASIC 2.0, a diferencia de otros lenguajes más evolucionados como el Pascal, no tiene funciones de líneas múltiples, ni variables del tipo booleano (que, sin duda, ayudarían mucho). Sin embargo la respuesta es un rotundo SI ¿Cómo hacerlo? Veamos primero cómo analiza el intérprete BASIC las distintas expresiones.

Cuando el BASIC se encuentra con sentencias del tipo:

```
IF A=B THEN ...
IF A$="PEPE" THEN...
IF K%<5 THEN ...
```

lo primero que hace es comparar lo que está a la izquierda del signo = (o < o >) , con lo que está a la derecha. Si la expresón es verdadera, el BASIC le asigna a la expresión el valor -1. Si es falsa, le asigna el valor 0. Si quieren probar, tipeen en modo



directo: A=(2=2):B(1>3):PRINT A,B

Aparecerá: -1 0

debido a que el BASIC le asignó a la primera expresión (verdadera) el valor -1, y a la segunda (falsa), el 0. El paso siguiente del BASIC es decidir qué hacer. Si el valor que le asignó a la expresión evaluada (la que está entre el IF y el THEN) es -1, ejecutará las instrucciones que sigan al THEN. Si por el contrario es 0, pasará a la siguiente. En otras palabras la decisión depende de si lo que está entre el IF y el THEN vale 0 (falso) o no vale 0 (verdadero). Por ejemplo, se puede probar colocando en modo directo:

```
IF 0 THEN PRINT "ESTO NO APARECERA"
IF 3 THEN PRINT "ESTO SI APARECERA"
```

Como dije más arriba, el BASIC 2.0 no tiene funciones de líneas múltiples. Es decir que deberemos arreglarnos con funciones de una sola línea ¿Y cómo funciona una función? Veamos:

Si queremos definir una funcion, debemos poner:

```
10 DEF FNA(X)=SIN(X)*COS(X)
```

Luego, en el programa, cada vez que queramos calcular el valor del seno por el coseno de una variable, bastará con poner: 100 PRINT FNA(FI).

Cuando el BASIC llegue a la línea 100, verá que debe calcular el valor de la función A para el valor FI.

Entonces pasará el valor de FI a la variable X. X es una variable de la función, es decir que, cualquiera sea el valor que tome X, no afectará el valor de otra variable X del programa principal.

Luego, calculará el valor de la función, y volverá al programa principal.

En una función se puede poner "cualquier cosa" que entregue un valor numérico, incluyendo instrucciones como PEEK(A), o expresiones como (A=B). Así, usando funciones, la expresión del principio puede reemplazarse por:

```
10 DEF FNJ2(X)=(((PEEK(56320)AND 16)=0) OR (PEEK(203)<>64))
```

La clave es poner en la función lo que esté entre el IF y el THEN (en la linea original).

Luego, cuando uno quiera saber si se presionó una tecla o el botón del joystick #2, bastará con poner:

```
100 IF FNJ2(0) then ...
```

En el lugar del 0 puede haber cualquier número, ya que la variable X no se usa realmente en la función.

Ventajas

1. Ahorro de memoria. Este será mayor cuanto más larga sea la expresión evaluada, o cuanto más veces se use en el programa.
2. Claridad. El programa resultante será más claro y fácil de entender. Para esto es recomendable darle a la función un nombre significativo (por ejemplo J1 para el joystick #1, J2 para el #2, etcétera).

Desventajas:

La principal es una pequeña pérdida de velocidad (cosa importante en un juego). En definitiva, es una técnica que se justifica sobre todo en los programas largos que ocupen gran cantidad de memoria.

Guillermo G. Roit

CONCURSO
de
NOTAS, TRUCOS y PROGRAMAS

Los trabajos pueden estar referidos tanto a la Drean-Commodore 64 como a la 128.
Las notas deben apuntar a un mejor aprovechamiento de las potencialidades de las máquinas.
Los trucos y los programas tienen que ser inéditos.
El software deben remitirlo a nombre de Concurso Revista para usuarios de Commodore, Paraná 720, 5º piso, (1017) Capital Federal, grabado en disquete o casete. Indiquen los datos del programa, computadora y autor. Es preferible (pero no imprescindible) que acompañen el listado del mismo por impresora.



un DRIVE Drean-Comm.



Menciones: 10 cartridges

CIERRE DEL PRIMER CERTAMEN: 27 de noviembre de 1987.

EL MICROPROCESADOR DE LA 64 (II PARTE)

Seguimos investigando, en esta ocasión, los registros que permiten el procesamiento de datos en el 6510.

En el número anterior hemos visto generalidades del 6510, diferenciando las líneas de datos, de direcciones y de control. Ahora nos dedicaremos a conocer los distintos registros que existen dentro del microprocesador.

STACK

El 6510 incorpora un registro que se ocupa de las operaciones de stack. El stack es una región de memoria de 256 bytes de longitud ubicadas entre las posiciones de memoria 256 y 511.

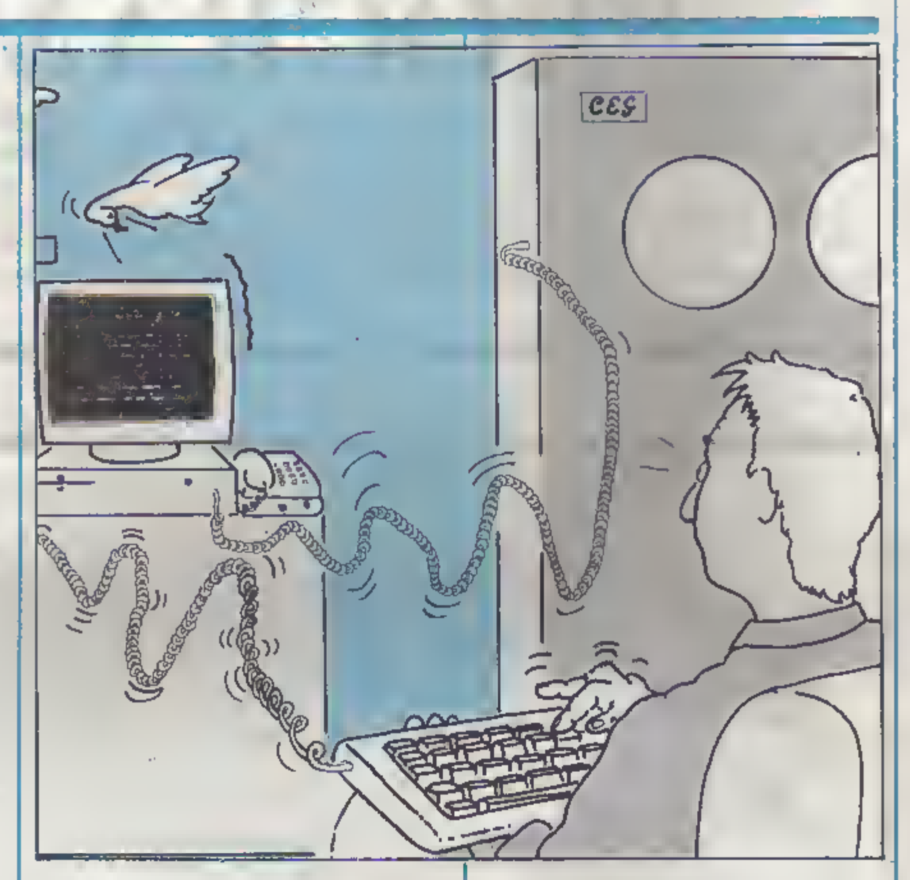
Si dividiéramos la memoria en 256 partes según el valor de los bits más significativos, obtendríamos 256 páginas de memoria, de manera que el stack ocuparia sólo la página 1. Ver figura 1.

¿Cuál es la función del stack?.

Es un lugar de almacenamiento provisorio de la información utilizado por el 6510 cuando debe postergar una operación por la llegada de una urgencia.

Imaginemos que nuestro trabajo es transportar bolsas de papas y que, de pronto, nos ofrecen una enorme torta de chocolate. Ante tan agradable urgencia seguramente dejariamos de lado las bolsas para comer la torta en un lugar tranquilo.

Una vez satisfechos regresaríamos para continuar con nuestro trabajo -llevar las bolsas-, pero muy probablemente no las encontraríamos. Lo mismo le ocurriría al microprocesador. Si él desechara la información anterior a la urgencia, cuando esta finalizase no tendría manera de reanudar su operación interrumpida. Por ese motivo se provee el uso de la región de stack: para almacenar todos los datos existentes en las regiones del microprocesador antes de atender la interrupción.



Para localizar luego esos datos preciosos el 6510 tiene un registro llamado Stack Pointer (puntero del Stack) donde se indica cuál es la ubicación de la última información almacenada como se observa en la figura 2.

Cuando la urgencia ya ha sido satisfecha, el microprocesador puede retomar el hilo de lo que estaba haciendo y recuperar su información del stack.

De la misma manera -en nuestro ejemplo- si antes de comer la torta hubiésemos colocado las bolsas en un casillero numerado, podríamos recuperarlas con sólo recordar en qué casillero las hemos dejado.

La única sutileza de esta operación consiste en recuperar los datos en el sentido inverso al que fueron depositados, es decir, tomar en primer término lo último que se almacenó. Esta forma de almacenar se llama estructura LIFO y se asemeja a una pila de libros como se muestra en la figura 3.

UNIDAD ARITMETICO LOGICA

En esta unidad se debate todo el fenómeno interior de la computadora. Una vez que el código de operación ha llegado y ha sido descompuesto y analizado dentro de micro, lo que queda es ejecutar la orden.

Muchas de estas órdenes tienen que ver con la unidad aritmético-lógica. Esta unidad es la única que realiza un procesamiento real de los datos. E-

Figura 1

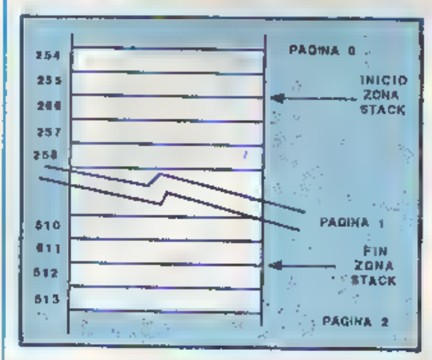
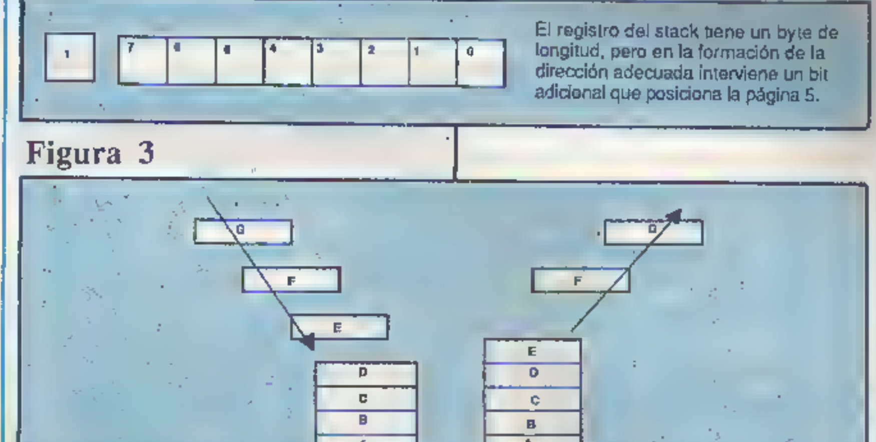


Figura 2



lla, a través de las distintas operaciones que tiene capacidad de hacer, modifica y crea nuevos datos.

Nuestra ALU tiene dos entradas y una única salida de 1 byte de extensión tal como se muestra en el esquema de la figura 4.

Dentro de las posibilidades aritméticas se encuentran: la suma de los bytes, la complementación de un byte y la posibilidad de sumar o restar 1 a sus regitros internos.

A partir de estas operaciones se logran todas las demás: resta, multiplicación y división.

Otro tipo de manipulaciones que la ALU es capaz de realizar son los desplazamientos y las rotaciones, que si bien están emparentadas, no son exactamente lo mismo. En estas operaciones se utiliza un bit adicional que se llama bit de Carry.

Los desplazamientos incorporan un "O" por una punta del registro mientras el exceso saliente de la otra punta se almacena en el mencionado bit de Carry. Si la memoria se repite, se incorporarán más "O" al registro, pero los bits desechados iniciarán un viaje sin retorno después de pasar por el bit de Carry.

Las rotaciones realizan el mismo tipo de movimiento lateral, pero con una variante: los bits que salen por una punta vuelven a ingresar por la otra -luego de atravesar al bit adicional- si la operación se repite más de dos veces. La rotación se parece mucho a una calesita.

En la siguiente figura se esquematiza la manipulación que realiza la ALU durante un desplazamiento y durante una rotación. Ver figura 5.

OPERACIONES LOGICAS

La ALU está habilitada para realizar algunas operaciones lógicas que se refieren al empleo de ciertas compuertas lógicas para procesar la información.

Estas operaciones son: AND,OR y OR-EXCLUSIVE.

Sin profundizar en el estudio de cada una de estas operaciones diremos que se ap lican a cada uno de los bits del byte a procesar.



A modo de ejemplo observaremos cómo resultan estas operaciones lógicas para una ALU de un solo bit. Ver figura 6.

La finalidad de estas operaciones es la misma que la de funciones aritméticas: generar nuevos números. Pero en este caso las operaciones OR y AND permiten la creación de "máscaras" que ocultan o muestran algunos bits del byte -dato con prescindencia de los otros, mientras que la OR-EXCLUSIVE brinda en la confrontación de dos bytes un "1" si los bits coinciden, y un "O" si son distintos.

ACUMULADOR

El acumulador es el registro más importante y versátil que se encuentra dentro del 6510.

Esto se debe a que muchas de las operaciones del set de instrucciones lo toman como protagonista exclusivo. Así, por ejemplo, la rotación de bytes que hemos visto o las comparaciones se realizan únicamente sobre el contenido de este registro especial.

La transferencia de información dentro de una computadora se realiza a través de un conjunto de líneas llamadas BUS de DATOS. De igual manera, dentro del chip existe un bus de datos interno que permite la comunicación entre los distintos registros sin demora, llevando la información a través de un sistema de ordenamiento de tráfico que abre y cierra compuertas. Estas compuertas están controladas por el INSTRUCTION REGISTER, (registro de Instrucciones).

La frontera entre el bus de datos interno y el externo es un conjunto de llaves electrónicas que no sólo controlan el paso de datos, sino que además establecen el sentido de circulación de los mismos. Los datos "entrarán" en el microprocesador cuando este desee tomar o leer información, (este trabajo se denomina operación de lectura). Por el contrario, los datos "saldrán" del chip cuando este quiera comunicar o almacenar algún resultado, en este caso la operación es de escritura.

El sentido de circulación depende del estado de la linea de control R/W*.

Ahora que ya conocemos el bus in-

terno podemos advertir lo siguiente: si más adelante analizamos alguna operación, como por ejemplo, la rotación del contenido del acumulador, sabremos que no es el acumulador quien realiza el trabajo, porque esta tarea sólo puede llevarla a cabo la ALU.

Lo que ocurre es que el acumulador le entrega a la ALU su contenido a través del bus interno. La ALU lo procesa y se lo devuelve con la modificación hecha por el mismo camino.

La importancia del Acumulador reside justamente en la cantidad de operaciones que realiza directamente con la ALU, facilidad que los otros registros no tienen.

REGISTROS INDICES

Bajo la denominación de registros indices distinguimos a otros dos que existen dentro del 6510: el registro X y el registro Y.

Son registros en todo el sentido de la palabra: pueden almacenar información provisoriamente o trabajar como auxiliares del acumulador; pero además son destinados para ciertas tareas que el presente o futuro programador en código de máquina habrá de valorar.

Estos registros, utilizados a través de algunas instrucciones, permiten almacenar valores que se suman o indexan a otros números para hallar una determinada ubicación.

Este mecanismo responde a una forma de direccionamiento llamado direccionamiento indexado.

A través de estos párrafos inferimos que, si bien es importante la cantidad de registros que un microprocesador posee, la potencia del mismo dependerá de la cal idad de sus instrucciones, (porque ellas son las que permiten sacarle el jugo a esos registros).

REGISTRO DE ESTADO

El registro de estado es un registro de 1 byte de longitud que almacena una información muy especial.

En él no encontraremos ni operandos ni códigos de operación -como ocurre con el acumulador o los registros X e Y- sino que su contenido se relaciona con aquello que está ocurriendo en la ALU y donde, además, cada uno de los bits que componen, el registro tiene un significado especial y distinto del de su compañero.

Figura 4

Para comenzar, digamos que sólo 7 de los ochos bits son utilizados. Estos bits son indicadores de proceso y pueden tomar el valor "1" o "O" según el resultado que la operación de la ALU arroje. Y justamente por señalar eso a estos bits indicadores se los llama FLAGS (banderas de estado). Veamos en la figura 7 cuáles son estas banderas y qué acontecimiento señalan.

Bit de Estado N: Este bit señala cuando, el contenido del acumulador en un número negativo. En ese caso el bit toma valor "1"; mientras el acumulador tenga un número positivo la bandera N permanecerá en "O".

Bit de Estado O: La letra O es la inicial de OVERFLOW, que significa desborde. Señala cuando luego de una operación se produce un exceso que lleva un "1" de la posición 6 del acumulador al bit 7.

Bit de Estado B: Este bit de estado indica cuándo se ha producido un Break durante la ejecución del programa, interrumpiéndolo.

Bit de Estado D: Este bit es una ayuda matemática destinada a facilitar las cuentas. Mientras el estado del bit D sea "O" todas las operaciones matemáticas se llevarán a cabo en la forma acostumbrada. Pero si toma el valor "1", las cuentas se realizarán en código BCD, que significa Decimal Codificado en Binario, (justamente D significa Decimal operation).

Figura 5

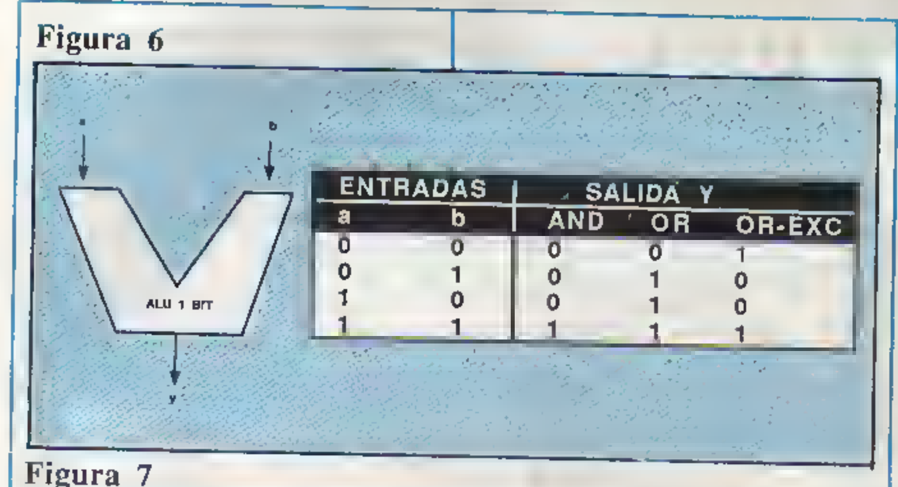


Bit de Estado I: Este indicador corresponde a una máscara que permite o no cierto tipo de interrupciones. Cuando el bit esté en "1" se dice que la interrupción está enmascarada y por lo tanto no puede ser interrumpido. El estado de este bit depende del programador.

Bit de Estado Z: Este bit señala con un "1" si el contenido del acumulador es cero. Mientras no lo sea permanecerá en "O".

Bit de Estado C: Este bit actúa como si fuera el noveno bit del acumulador almacenando algún eventual exceso de una operación matemática

Figura 6



ENTRADAS		SALIDA Y		
a	b	AND	OR	OR-EXC
0	0	0	0	1
0	1	0	1	0
1	0	0	1	0
1	1	1	1	1

Figura 7

ACUMULADOR

REGISTRO X

REGISTRO Y

REGISTRO DE ESTADO: N | O | | B | D | IRQ | Z | C

o interviniendo en las operaciones de rotación ya vistas.

Ahora que hemos tomado contacto con los distintos tipos de bits de estado podemos preguntarnos sinceramente:¿Cuál es la utilidad de estos registros?

Antes que nada debemos decir que estos bits no permanecen estáticos durante la ejecución de un programa.

Es más, si cada uno de los indicadores tuviera asociada una lamparita - de manera tal que esta se encendiera al estar en "1"- veríamos un titilar permanente. Porque, como dijimos, el estado de las banderas depende de las operaciones que se llevan a cabo en la ALU y de los resultados que ella arroje.

Veamos lo siguiente: las computadoras no sólo deben ser capaces -como lo son- de efectuar operaciones matemáticas a altisíma velocidad, sino que deben ser capaces de tomar decisiones. Si esto no fuera así, no sería posible la ejecución de ese conjunto de instrucciones que conocemos como programa. Porque a la primera disyuntiva presentada ...pifff!!! se nos pincharía la máquina. Imaginemos lo siguiente: si en BASIC hiciéramos un simple loop, para demorar unos segundos (del tipo FOR-NEXT) y no hubiera elemento capaz de detectar que la condición de finalización ha sido satisfecha, la máquina podría seguir años en ese loop hasta que una mano samaritana la desconectara.

Para salvar ese escollo se hizo imprescindible crear algo que hiciera posible la toma de decisiones. ¿Y saben quiénes son la respuesta a este problema?, SI!!! los indicadores de estado.

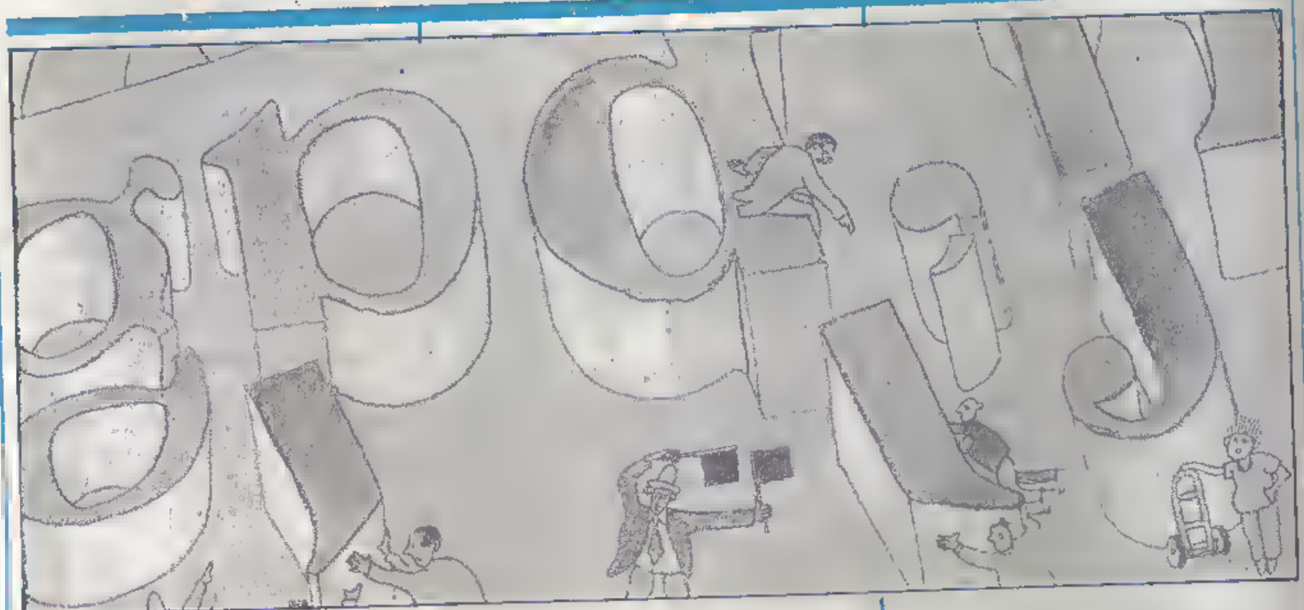
Porque, en forma similar a lo visto en el ejemplo en BASIC, en el momento de tomar alguna decisión durante el programa lo que se hace es examinar el estado de alguno de los indicadores. Si la condición no es satisfecha, se sigue adelante,de lo contrario se ordenará una bifurcación que le permitirá al 6510 continuar con otra tarea.

Así es como estos minúsculos auxiliares que habitan dentro del microprocesador permiten la inteligencia mecánica que tanto ahorro de trabajo y momentos de esparcimiento le procuran al hombre. De esta manera culminamos la excursión a los dominios del 6510.

Guillermo Fornaresio

CARACTERES RAM

Autor: Gerardo Krizan
Tipo: Utilitario
Comp.: DreanCommodore 128



Copiemos la totalidad del juego de caracteres de la ROM a la RAM, en una zona libre (entre 12228 y 16383) de manera que podamos redefinirlo sin interferir con los programas BASIC ni con las variables.

REDEFINICION DE CARACTERES

El primer paso consiste en diseñar los modelos de los nuevos caracteres. Para ello es necesario saber cómo se representan estos en la pantalla. Un carácter se representa por una cuadrícula o matriz de 8*8 puntos.

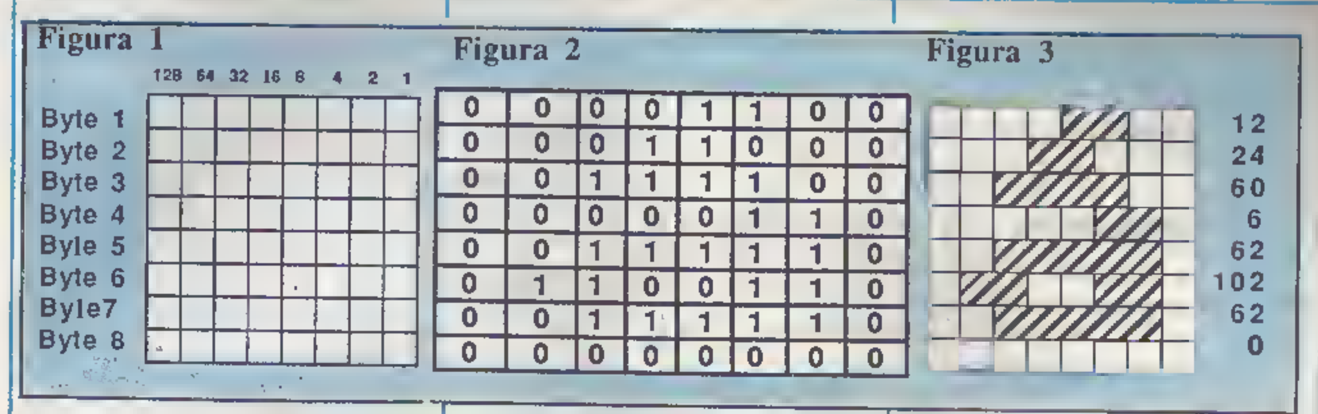
Cada fila de la cuadrícula corresponde a un byte, de tal manera que se precisan 8 bytes para definir un carácter. A su vez, para cada columna corresponde un valor (figura 1).
Cada punto puede presentar dos posibles estados: "1" si está activado, o "0" si esta apagado. En la figura 2, por ejemplo, se muestran los puntos encendidos- apagados correspondientes a la letra "a" minúscula acentuada.

Para cada byte se suman los valores correspondientes a las columnas que presentan el punto activado, obteniéndose el valor en decimal de dicho byte. Ese valor puede ir desde 0 (todos los puntos apagados) hasta 255 (todos activados) (ver figura 3). De esta manera cada carácter se representa mediante 8 códigos que se almacenan como 8 bytes en la tabla del juego de caracteres. Al trasladar la tabla de la ROM a la RAM (líneas 5025 a 5050), es posible modificar la información contenida en el generador de caracteres, y así definir nuevos símbolos en reemplazo de los caracteres estándar. Para ello, una vez calculados los valores de cada byte de cada nuevo símbolo se debe elegir el carácter estándar al que reemplazará, y el set a que corresponderá (1 o 2).
Supongamos que deseamos trabajar en el modo mayúsculas-minúsculas (set 2) y deseamos incorporar la letra a minúscula acentuada (á). Entonces podemos remplazar el símbolo "X" (que se obtiene al pulsar las teclas

ESTRUCTURA DEL PROGRAMA

LINEAS	FUNCION
5000-5010	Pantalla de presentación.
5015 (*)	Pasa a modo 2 MHZ.
5020 (*)	Limpia la pantalla.
5025-5050 (*)	Pasa el juego de caracteres de ROM a RAM.
5055 (*)	Pasa a modo 1 MHZ.
5060 (*)	Indica al ordenador el uso de la tabla de caracteres de la RAM.
5065-5075 (*)	Coloca en memoria los valores de los nuevos caracteres.
5080 (•)	Pasa a modo mayúsculas/minúsculas.
5085-5175	Imprime en pantalla la tabla con los nuevos caracteres y la forma de obtenerlos.
5180 (°)	Vuelta de la subrutina.
5200-5310 (*)	Datos de los nuevos caracteres.

(*) Líneas imprescindibles para el funcionamiento del programa. Las restantes puede omitirse o modificarse.
(•) Se omite si se desea trabajar con el set número 1.
(°) Presente solamente si este programa constituye una subrutina de otro programa



Commodore y A simultáneamente) por el nuevo carácter diseñado. Esto se realiza haciendo el POKE de los 8 nuevos valores de los bytes en las direcciones de memoria RAM correspondientes (líneas 5065 a 5075).

¿Cómo se calcula la dirección de comienzo de un carácter? Las direcciones del juego de caracteres en la RAM están comprendidas entre 12228 y 14335 para el set 1 y entre 14336 y 16383 para el 2. Para conocer la dirección de inicio de un carácter es necesario contar con la tabla de códigos de pantalla (screen display codes) que se encuentra en el manual del usuario. Dicha tabla cuenta con tres columnas: la primera muestra los caracteres correspondientes al set 1, la segunda a los del 2 y la tercera indica el orden de almacenamiento de cada carácter en el juego de caracteres (columna de poke).

Por ejemplo, para conocer la dirección de inicio del carácter "X" en el set 2, buscamos en dicha tabla y encontramos el número 112. Multiplicando este número por 8 obtenemos el número de byte de comienzo de este carácter, 896, dentro de los 2048 bytes que constituyen el juego de caracteres de este set. Sumando 896 a la dirección de inicio del set de caracteres en RAM (en este caso14336), da 15232 que es la dirección de inicio del carácter "X".

El último caso consiste en crear las líneas DATA con los valores de dirección de memoria y bytes del carácter a redefinir. Si se desea redefinir también el carácter inverso, su dirección de inicio se obtiene de sumar 1024 al valor correspondiente al carácter normal. Los valores de los bytes serán el complemento de 255 de los valores del carácter normal (255 menos el valor del carácter normal).

En el programa se redefinen 12 caracteres: á, é, í, ó, ú, ü, ñ, Ñ, ª, º, ¿ y ¡.

Este trabajo puede servir como base para cualquier redefinición de caracteres, con solo modificar la línea 5065 (cambiando el valor incial del lazo por el número de caracteres menos uno a redefinir) y colocar los DATA correspondientes.

VARIABLES UTILIZADAS

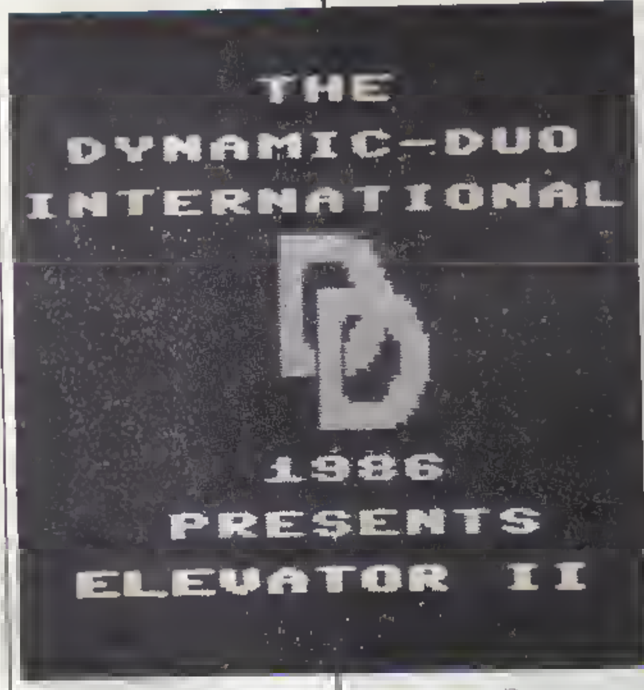
A: Contiene número del byte dentro del juego de caracteres.
B: Contiene los valores de los bytes del juego estándar de caracteres.
A 1: Contiene el número de caracteres a redefinir.
C: Contiene la dirección de inicio de cada carácter.
A2: Contiene el número de byte de cada carácter.
D: Contiene el valor de cada byte del nuevo carácter.

```
10 REM ****************************
20 REM *                          *
30 REM *    CARACTERES RAM        *
40 REM *                          *
50 REM *(C) 1987 * COMMODORE*
60 REM *                          *
70 REM *    GERARDO KRIZAN        *
80 REM *                          *
90 REM ****************************
100 ::
5000 PRINT"        ESPERE 45 SEGUNDOS"
5005 PRINT"      C A R G A N D O   D A T O S"
5010 SLEEP2
5015 FAST
5020 SCNCLR
5025 BANK14
5030 FORA=0TO4095
5035 B=PEEK(53248+A)
5040 POKE12288+A,B
5045 NEXT
5050 BANK15
5055 SLOW
5060 POKE2604,28
5065 FORA1=0TO11:READC
5070 FORA2=0TO7:READD:POKEC+A2,D
5075 NEXT:NEXT
5080 PRINTCHR$(14)
5085 PRINT"     NUEVO SET DE CARACTERES."
5090 PRINT"   "
5095 PRINTTAB(9)"
5100 PRINTTAB(9)"
5105 PRINTTAB(9)"
5110 PRINTTAB(9)"
5115 PRINTTAB(9)"
5120 PRINTTAB(9)"
5125 PRINT"
5130 PRINT"          SIGNIFICA QUE SE DEBE"
5135 PRINT"          PULSAR LA TECLA "
5140 PRINT"          A LA VEZ QUE SE PULSA"
5145 PRINT"          LA  TECLA CORRESPONDIEN-"
5150 PRINT"          TE."
5155 B$="PULSE UNA TECLA -- "
5160 PRINT"                   "SPC(10)B$
5165 B$=MID$(B$,2,18)+LEFT$(B$,1)
5170 FORT=1TO100:NEXT
5175 GETA$:IFA$=""THEN5160
5180 END
5200 DATA 15160,60,0,102,118,126,110,102,0
5210 DATA 15184,60,0,124,102,102,102,102,0
5220 DATA 15232,12,24,60,6,62,102,62,0
5230 DATA 15240,12,24,60,102,126,96,62,0
5240 DATA 15120,12,24,0,56,24,24,60,0
5250 DATA 15304,12,24,60,102,102,102,60,0
5260 DATA 15296,12,24,102,102,102,102,60,0
5270 DATA 15152,24,0,0,24,24,24,24,0
5280 DATA 15072,24,0,24,48,96,102,60,0
5290 DATA 15168,102,0,102,102,102,102,60,0
5300 DATA 15136,24,60,102,255,102,255,102,0
5310 DATA 15096,48,72,72,48,0,120,0,0
```

★★★★★	EXCELENTE	★★	REGULAR
★★★★	MUY BUENO	★	MALO
★★★	BUENO		

ELEVATOR ACTION DOS

Comp.: Drean Commodore 64/C
Editor: Dinamic Duo
Distribuye: The Tuerk
★★★



Una nueva versión del juego del ascensor, en la cual nuestro hombre ingresará a un rascacielos plagado de peligros y agentes enemigos por el piso 30. Deberá descender hasta sus profundidades, evitando los disparos de los adversarios, pasando de ascensores a montacargas y, en otras oportunidades descendiendo, simplemente, por las escaleras.
Al comenzar el juego dispondremos de tres vidas. Cada enemigo eliminado son 100 puntos y, alternativamente, algunas de las puertas de los muchos departamentos del rascacielos comenzarán a brillar. Ese será el momento indicado para ingresar y salir de una de ellas y acumular 200 puntos.
Para evitar los disparos de los enemigos, nuestro hombre podrá agacharse -tratando con sus buenos reflejos que los disparos pasen por sobre su cabeza-, o por el contrario saltar hacia adelante y arriba, para quedar más cerca del hombre de impermeable y sombrero que nos dispara y así poder eliminarlo.
Particularmente difícil será resolver el tramo que comienza en el fin del recorrido del primer ascensor, y donde están los montacargas de funcionamiento constante, ya que en su piso más bajo nos esperarán dos enemigos, uno de los cuales -si no aprovechamos al máximo nuestros reflejos- nos matará.

Mucho cuidado, eso si, con caerse por los huecos de los ascensores o con llegar con ellos hasta el fondo de todo, si no quieren pasar a mejor vida.

EPIX STARFIRE

Comp.: Drean Commodore 64/C
Editor: Non Soft
Distribuye: Papillón
★★★

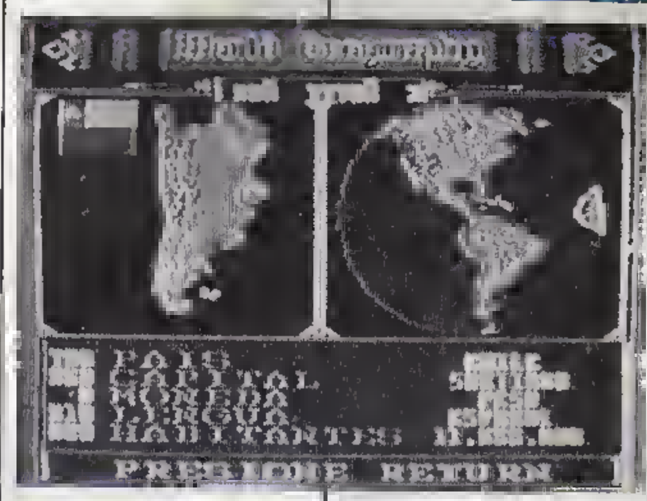
La Guerra de las Galaxias sigue en pie, y ahora debemos combatir contra numerosas naves enemigas que, con sorprendente capacidad de maniobra, aparecen en nuestra zona de patrulla y nos dispararán con sus cohetes láser.
Sentados en la cabina de comandos de nuestro caza galáctico, tenemos una visión del espacio ubicado a nuestro frente, mientras en la parte inferior de la pantalla disponemos de un radar -para ubicar las naves enemigas y dirigirnos hacia ellas- y un indicador sobre el nivel de energía de nuestras armas láser.
Conviene evitar la tentación de mantener el botón de fuego constantemente apretado, porque corremos el riesgo de, una vez que centremos sobre la mira al enemigo, carezcamos de energía y debamos esperar preciosos segundos para hacer el disparo, dejando escapar a la presa.
Cuatro diferentes tipos de naves se nos enfrentarán: las tres primeras -similares a las de la película La Guerra de las Galaxias, son de color azul, verde y rojo, cada una de ellas destruida representan, respectivamente 10, 20 y 40 puntos.
La cuarta es nuy parecida a los transbordadores espaciales y, si logramos acertarle, ganaremos 50 puntos.
El juego tiene 10 niveles ("0" el más fácil y "9" el de mayor complejidad), directamente vinculados con la agilidad de los enemigos y la capacidad de recarga de nuestro láser.
Si seleccionamos el nivel cero, por ejemplo, dispondremos de un volúmen de combustible predeterminado y estaremos obligados a alcanzar los 300 puntos para volver a obtener un tanque completo. Claro que, como nada es gratis en esta vida, por el carburante adicional deremos pagar como precio el pasar al nivel uno, y así sucesivamente hasta el último de ellos.

WORLD GEOGRAPHY

Comp.: DC 64/C
Distribuye: Centro de Atención al Usuario
★★★★★

La enseñanza asistida por computadora se basa en software elaborado para introducir al educando en universos nuevos, invitándolo a explorar el tema desarrollado acompañado por un programa y asistido por el maestro.
Los tipos de programas educativos son variados, para cubrir la gama de objetivos del educador.
De esta manera nos encontramos con programas tutoriales, simuladores, utilitarios, juegos lógicos y de ejercitación según la clasificación habitual.
Pero muchas veces no es posible aplicar esta clasificación para un programa particular ya que el mismo puede ser incluido en más de un tipo.
Uno de estos casos es el **World Geography**. Si bien su nombre figura en inglés, el resto de la información está en nuestra lengua.

REVISION DE SOFTWARE



El programa puede ser usado como profesor o tutor, que nos detallará los aspectos de la geografía política de una determinada región o país, se puede utilizar como consulta y también tiene una opción de juego para facilitar la ejercitación y fijación de conocimientos. Una vez hecha la elección, el globo terráqueo que está en pantalla comienza a dar vueltas hasta estacionarse en el continente donde se halla el país elegido, y el lugar donde está ubicada la capital empieza a titilar. Además se brinda una detallada información que incluye el nombre del país, la capital, moneda, lengua, cantidad de habitantes y la bandera de la nación.

Con estos datos a su disposición los niños pueden utilizarlos para tareas requeridas por sus maestros, como demuestra la experiencia del Centro de Atención al Usuario Drean Commodore.

Allí se realizó una demostración de este educativo ante alumnos de nivel primario, notándose de inmediato la atracción que los niños sentían por el programa debido a su facilidad de uso y su aplicación inmediata, ya que tenían que buscar y dibujar las banderas de varios países. Pero este programa cuenta aún con otra posibilidad, que es la ejercitación, a manera de juego, que permite fijar los datos más importantes de cada país.

Esto se hace mediante el método de elección múltiple



para cada uno de los ítems mencionados anteriormente.

Se puede elegir el grado de dificultad entre dos niveles, para que las elecciones de los países y las propuestas sean más o menos difíciles.

Ante cada respuesta correcta una serie de notas musicales son ejecutadas a manera de aprobación, y cambian de tono si la respuesta indicada es incorrecta. En este caso se le da al alumno la contestación y se sigue con el próximo ítem.

Este verdadero atlas universal computarizado permite una interacción entre niño y computadora y, bien utilizado, cubre varios objetivos de la enseñanza asistida por computadora en materia de información brindada, ejercitación y fomento de la investigación.

En síntesis tenemos aquí un programa educativo para la Drean Commodore 64/C, o la 128 en modo 64, de un muy buen nivel técnico y didáctico, lo que lo hace muy recomendable para que los niños tomen contacto con la geografía política universal.

HOME PACK

Comp.: Drean Commodore 64/C
Edita: Star Systems Software
★★★★

El **HomePak** es un programa para la Drean Commodore 64 que el desconocimiento y la falta de desarrollo de la telemática tiró a un costado durante tres años. Se trata de un **integrado** que consta de procesador de textos, base de datos y terminal de telecomunicaciones diseñado por Russ Wetmore y editado por la Star Systems Software Inc. de Canadá en 1984. Tiene un total de 418 bloques de información, a razón de unos 120 por programa principal, a los que se debe agregar la presentación con el menú principal y cuatro suplementarios para configurar los valores defaults propios, transferir archivos entre las diferentes opciones del trío central, copiador de cara entera, y un compresor para los registros con información.

El procesador de texto, llamado **HomeText**, en cuanto a funciones básicas se refiere, tiene poco que envidiarle a cualquiera de sus pares. Es de muy sencillo manejo, tiene letras redefinidas de cuerpo muy grueso y constantemente. En la parte inferior de la pantalla, escribe leyendas para ir a aquella con el menú de funciones, el remanente de memoria en cantidades de caracteres, qué función está activa y si la impresora está o no presente. Su capacidad total es de un poco más de 13,5 K, o sea, unas diez carillas carta en la vieja máquina de escribir. Como es obvio, mediante el **HomeTran** que convierte archivos para usar los que generan los otros, es posible hacer **merge** con lo acumulado en la base de datos así como listados para **mailing**. Una de las características sobresalientes, bastante poco habitual en procesadores para 64K, es la posibilidad de ver previamente en pantalla al texto tal como va a salir impreso. De esa forma se pueden hacer las correcciones y alteraciones necesarias en el formato final. La otra es la

alternativa del envío telemático de un texto allí elaborado, así como la recepción por la misma vía y su posterior edición en el procesador.

El primer aporte realmente original y excepcional de este **integrado** para la C-64 aparece en **HomeFind**, su base de datos. Si bien cada registro puede tener un máximo de sólo 144 caracteres, lo que puede dar una apariencia de excesiva limitación; las caracteristicas con que viene dotado le otorgan una funcionalidad y utilidad realmente inéditas.

Primero que nada, en el diseño del registro no se hace necesario marcar campos en el sentido conocido del término. La información debe ser ingresada "telegráficamente" dividida de la siguiente forma:

ASUNTO (subjetct) IDENTIFICATION (tag) OBJETIVO (object)

Cada una de estas partes de la "oración" que exige el **HomeFind** no puede tener más de 80 caracteres de largo, pero en total, como ya vimos, no puede exceder los 144. Además los dos primeros deben ser señalados al final de cada uno, para que la máquina no los confunda, con un apóstrofe **y una s.** Ejemplo:

PROEDI's edita's mensuario DREAN COMMODORE

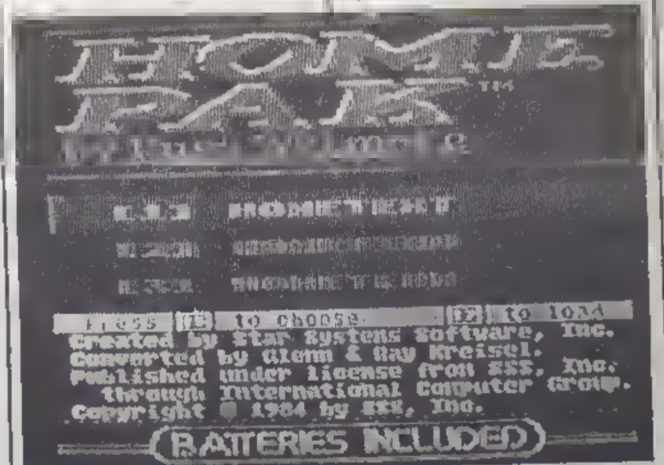
El **HomeFind** automáticamente va creando tres índices simúltaneos: uno para el ASUNTO, otro para la IDENTIFICACION y el tercero para los OBJETIVOS. En cualquier momento se puede requerir información en cualquiera de ellos, de diversas maneras, como veremos enseguida, y aquí es por donde empieza a mostrar las múltiples, muy prácticas y utilísimas aplicaciones esta base de datos. Vamos a seguir con los ejemplos:

PROEDI's edita's mensuario K64
PROEDI's teléfono's 49-7130
PROEDI's teléfono's 46-2886
PROEDI's radiollamada's 311-0056
PROEDI's dirección's Paraná 720, 5, Capital Federal.
PROEDI's edita's publicaciones de informática.

Antes de continuar, siempre ateniéndose a los ejemplos, IDENTIFICACION y OBJETIVO. Ahora, si se le pregunta:
Where Proedi's dirección's, no tardará en aparecer **Paraná 720, 5, Capital Federal.**

Las combinaciones pueden ser muchas. Ejemplo: si le preguntamos **What's Paraná 720, 5, Capital Federal?**, nos responderá **Proedi's dirección's.**

Estas aptitudes convierten a **HomeFind** en una base de datos más que apta para te-



el requerimiento se puede hacer por **PROEDI, Proedi o proedi** ya que el programa lo reconoce sin problemas. Lo que no se puede hacer es usar una misma palabra o concepto de varias palabras como ASUNTO y después como IDENTIFICACION o viceversa. Tampoco uno se puede olvidar de señalar el fin de cada uno de estos términos en la sintáxis con 's para que el programa sepa de qué se trata. Otra regla inviolable es que cada ingreso debe contar con las tres partes señaladas. Las preguntas para una búsqueda se hacen con las palabras inglesas **WHAT, WHO, WHERE** y **WHEN.** Por ejemplo, si se le pregunta:
Who's Proedi's?, no tardará, en responder lo que edita, teléfonos, radiollamada, dirección y todo lo que ese ASUNTO tenga como

ner a mano del procesador de texto con glosarios, diccionarios con terminología específica, agendas de teléfonos, direcciones y cualquier otro archivo de datos que se pueda sintetizar en menos de 144 caracteres. Para acelerar la búsqueda y ahorrar espacio, el **HomePak** trae como programa suplementario un compresor que se pide desde el menú de funciones en la base de datos. Por último, tenemos el **HomeTerm**, nombre con que se identifica al programa de telecomunicaciones que convierte a la máquina en una terminal. Permite comunicarse tanto en 300 como en 1200 baudios. Para la recepción o envíos de programas, esto puede suceder en código ASCII, Modem o Vidtext, que es la característica especial de control que usan servicios como **CompuServe.** También, en el caso de comunicación máquina-a-máquina con otra C-64, el **HomeTerm** envía y recepta códigos, caracteres invertidos y otros símbolos especiales del PET ASCII.

En los que hace a los modems, como el programa ignora totalmente las paridades, se puede usar prácticamente con todos y tanto en sistema Bell como CCIT.

El **HomeTerm** consta de dos pantallas. Una es la llamada interactiva, que es donde se visualizan las alternativas de la transmisión, y la otra, a la cual se accede con F5, es la de menú de funciones. La memoria buffer tiene una capacidad máxima de 11,2 K, y al pie de la pantalla constantemente hay líneas de status que indican la velocidad en que se está operando, si es en Half o Full, el tipo de código y un reloj que nos indica el tiempo de uso. Por supuesto, desde el menú de funciones se puede hacer un **dump** del buffer por impresora y entre su aptitudes se encuentra la del autodiscado.

Entre sus virtudes, **HomePak** no cuenta -todavía- con un manual en castellano. El que circula es el original inglés. Ahora, como se trata de un integrado que realmente no ofrece en el manejo mayores dificultades que las comunes, tanto las pantallas con menú como las síntesis con la totalidad de comandos para el procesador de texto, la base de datos y el programa terminal que trae el manual permiten superar el bache sin problemas insalvables. En suma, estamos frente a un utilitario justo para estos tiempos que corren, caracterizados por la telemática, después de haber estado arrumbado durante tres años sin que nadie le prestara atención.

RANKING DE SOFTWARE

Por votación de los lectores se elegirán los cinco mejores programas de juegos o utilitarios creados para la computadora Drean Commodore 64C.
Para participar se debe enviar el cupón (o fotocopia) a nuestra Re dacción: Paraná 720, 5° Piso, Cap. Fed. (1017), personalmente o por correo.







1º- 1942
2º- COMMANDO
3º- GHOST'N GOBLINS
4º- INTERNATIONAL KARATE
5º- URIDIUM

PREMIO: 10 CASETES - 1 por ganador

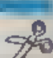
Los ganadores deel concurso de ranking de softwarc son:
Juan Pablo Lucena, Mendoza. Hernán Julita Leporace, Capital. Luciano Supertino, Córdoba. Gerald Meyer, Misiones. Carlos Cal, Santa Fe. Pablo Luis Pérez, Capital. Luciano Capobianco, Rosario. Guillermo Pipet, Córdoba. Juan Pablo Kutianski, Bragado (Bs. As.). ignacio Lafferriere, Capital.



Entre los que envíen sus cupones sc sortearán mensualmentc:

Diez Software para Drcan Commodore 64C

El franqueo para enviar premios es por contrareembolso a pagar en destino (o similar).



CUPON RANKING DE SOFTWARE DREAN COMMODORE

APELLIDO .... NOMBRE ....
DIRECCION .... LOCALIDAD ....
PROVINCIA .... C.P. .... T.E. ....
OCUPACION .... EDAD ....
COMPUTADORA ....

1° ....
2° ....
3° ....
4° ....
5° ....

QUE ES LO QUE MAS ME GUSTA DE LA REVISTA DEL USUARIO DE DREAN COMMODORE
....
QUE ES LO QUE MENOS ME GUSTA ....
....

AMPLIADOR DE CARACTERES

Revisando números atrasados de la revista "descubrí" en el número 8 un programa que se llama "Ampliador de Caracteres" que me pareció muy interesante.
Tuve sin embargo problemas en el momento de cargarlo, ya que una vez copiado y salvado el listado uno, al ejecutarlo me da DATA ERROR. Modifiqué algunas cosas y algunas veces me da DATA OK y otras me sigue dando error. ¿Podrían explicar cómo cargarlo?

Waldemar Schneider
Capital Federal



Los datos que nos das son pocos pero sirven para deducir qué tipo de error se produjo. Entre las líneas 160 y 190 se leen todos los datos, y hay una verificación (línea 200) de que se han ingresado los datos correctos.
Un tipo de error podrá haber sido la omisión o cambio accidental de algún dato. Pero por lo que nos cuentas, como alguna vez dio el DATA OK, parece que el error está en otra parte. Lo más probable es que te hayas confundido y puesto una "O" en donde debía ir "D", o un "8" por una "B".
Justamente tienes que hacer una copia de cada versión que haces (recordemos que este es un programa que se "autodestruye"), y guardar aquella que haya dado DATA OK.
Una vez hecho esto, ejecútalo y carga el listado 2, que es una demostracion del funcionamiento del programa. Ejecutándolo en pantalla veremos escrito bien grande "AMPLIADOR DE CARACTERES DREAN COMMODORE".

Continuamos con esta sección para que los lectores planteen sus consultas y sugerencias. Para eso deben escribir a Revista para usuarios de Drean Commodore, Paraná 720, 5to. Piso, (1017) Cap.

ROSARIO

Quisiera ponerme en contacto con usuarios de Drean Commodore aquí en la ciudad de Rosario

ARIEL P. CARAMUTO
Richieri 730 2000 - Rosario

La direccion del Centro de Atención al Usuario Drean Commodore en la ciudad de Rosario es San Martín 641.

INTERCAMBIO DE PROGRAMAS

Quiero felicitarlos por la revista, que viene superándose constantemente.
Tengo aproximadamente 300 programas y me gustaría intercambiarlos. Mi teléfono es 28-6490.
Tengo más de 150 programas en casete y me gustaría pasarlos a disco, ¿hay algún copiador que me sirva?

FARIAN BERNED
Capital Federal

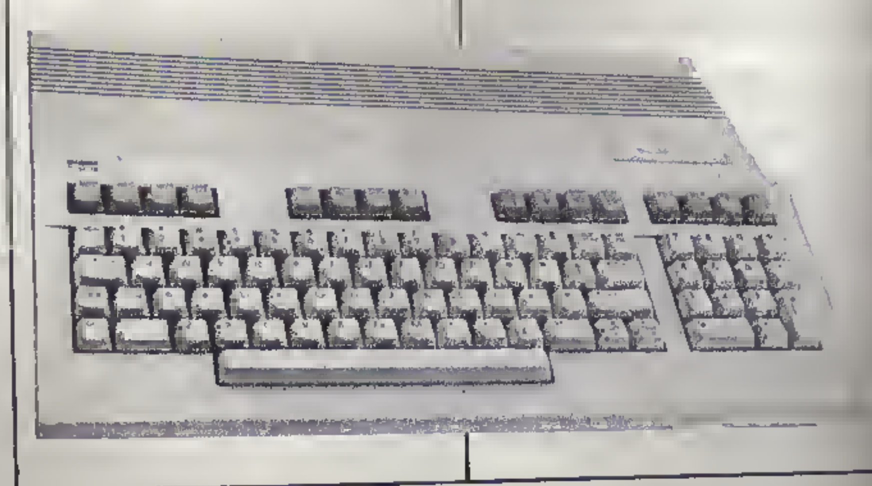
Publicamos tu teléfono para que los interesados puedan ponerse en contacto contigo.
En cuanto a los copiadores, está el COPY 190, pero el mismo puede "bajar" a casete muy pocos programas comerciales debido a las protecciones de los mismos. Para los programas que tú haces no tienes problemas para usarlo.
Otro copiador que posee la opción de disco a casete es el TURBO.COM.

INPUT AT

En la nota que publicaron en el número de septiembre de este año sobre el PRINT AT e INPUT AT para la C-128, mencionan la posibilidad de utilizar el comando DO... LOOP... WHILE o UNIL. ¿Podrían ampliar o hacer una nota sobre este comando?

JUAN C. CORONEL
Rosario

El DO-LOOP es un tipo de lazo que está presente en otros lenguajes además del BASIC. El "LOOP" cumple la función del NEXT en el FOR-NEXT. El DO indica el comienzo del proceso iteractivo que se ejecuta mientras se cumpla una condición determinada (DO WHILE condición). La alternativa es que el proceso se ejecute hasta que se de la condición de finalización (DO UNTIL condición de finalización. Por ejemplo una variable cuya suma supere un valor dado.



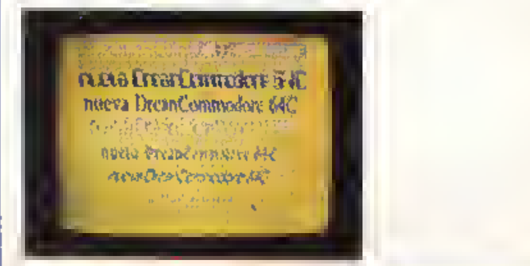
Drean C= COMMODORE 64C

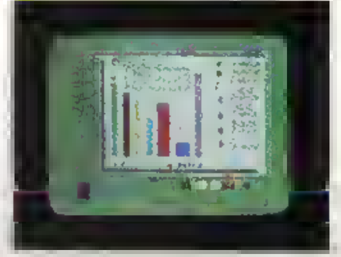
LA COMPUTADORA PERSONAL MAS VENDIDA DEL MUNDO!!



AHORA CON MAS PRESTACIONES!!

LA NUEVA DREAN COMMODORE 64 C INCORPORA EL PROGRAMA MAS NOVEDOSO
DE DIBUJO Y COMPOSICION DE TEXTOS.
ESCRIBE Y EDITA EN PANTALLA.
SELECCIONA 6 DIFERENTES TIPOS DE LETRAS EN 6 MEDIDAS DISTINTAS.
LE PERMITE DIBUJAR, PINTAR Y BORRAR EN PANTALLA.
DISEÑA CON 32 PATRONES.
PINTA EN 16 COLORES.





LA ULTIMA PALABRA EN TELECOMUNICACIONES

CON SU NUEVA DREAN COMMODORE 64 C,
PROVISTA DE UN MODEM, USTED PUEDE COMUNICARSE,
CON EL PAIS Y EL MUNDO MEDIANTE
EL 1º SERVICIO ARGENTINO
DE INFORMACIONES Y COMUNICACIONES
EN LINEA (DELPHI).
ADEMAS LE PERMITE INTERCAMBIAR
MENSAJES CON AMIGOS Y EL CLUB DE USUARIOS DREAN
COMMODORE, CON 25 FILIALES EN TODO EL PAIS
QUE LE BRINDARAN EL ASESORAMIENTO QUE USTED NECESITA.
ESTAS SON SOLO ALGUNAS COSAS
QUE USTED PUEDE HACER CON LA
NUEVA DREAN COMMODORE 64 C.

FABRICADO POR Drean SAN LUIS S.A.

A LA VANGUARDIA DE LA INFORMATICA EN ARGENTINA